www.freesoftwaremagazine.com

Issue 2 - March, 2005

Free Software MAGAZINE

FOCUS

FIGHTING SPAM

(KIRK STRAUSER)

THE HISTORY AND FUTURE OF SMTP

(KIRK STRAUSER)

FILTER SPAM WITH POSTFIX

(JOHN LOCKE)

HOW TO GET DSPAM, POSTFIX AND PROCMAIL TO PLAY WELL TOGETHER

(CHRIS J. KARR)

Poking at iTunes

A DEVELOPER'S GUIDE TO THE ITUNES PLATFORM

(AARON KROWNE)

THE FUD-BASED ENCYCLOPEDIA

DISMANTLING FEAR, UNCERTAINTY, AND DOUBT, AIMED AT WIKIPEDIA AND OTHER FREE KNOWLEDGE RESOURCES

(WILL WINKELSTEIN)

Why free IT management tools are gaining traction

ENTERPRISES ARE INCREASINGLY RECEPTIVE TO FREE SOFTWARE ALTERNATIVES FOR IT MANAGEMENT

(TOM CHANCE)

Guerrilla marketing

PART ONE: PROMOTING COMMUNITY PROJECTS IN THE MARKETPLACE

qreate

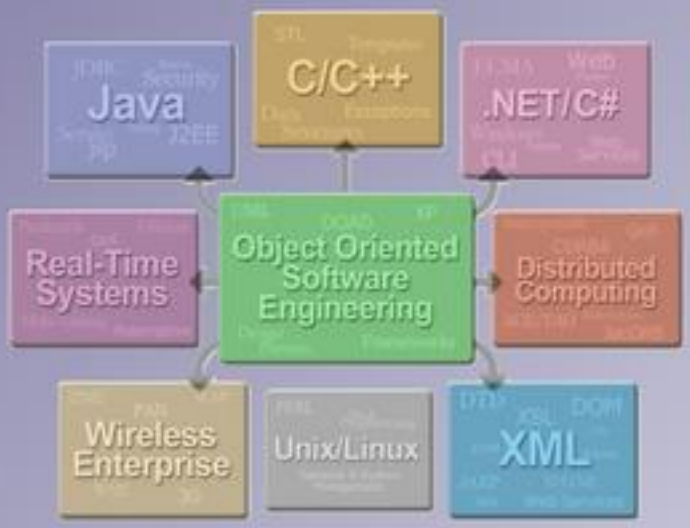
COVER BY WWW.QREATE.IT

Training for the Software Engineering Community

OCI, leading supplier of Object Oriented technology training, has been providing educational services for over 12 years.

Through our extensive experience in software training and consulting, our curricula has been refined utilizing open source products whenever possible to demonstrate concepts (**"concepts endure, tools change"** is our philosophy).

OCI's extensive curricula of over 50 hands-on workshops is delivered to corporations and individuals throughout the U.S. and internationally.



To learn more about OCI's

Educational Services,

visit us at **www.ociweb.com**

Curriculum-Based, Comprehensive, Pragmatic
- Standards-based/vendor-neutral
- Modular and Progressive
- Real-world examples

Customized Training Programs
- Skill/Gap Analysis
- Training Plans

Re-Skill Existing Developers
- Intro, Intermediate, and Adv. Levels
- OO-transitional curriculum

Knowledge Transfer Professionals
- Active practitioners/mentors
- Excellent client ratings

Integrated Mentoring
- Reinforcement following training
- Investment protection

Flexible Delivery Models
- On/Off Site
- Day/Evening
- Public/Private

Product Development ▫ Consulting ▫ Educational Services

Object Computing, Inc. (OCI)

St. Louis, MO Headquarters: +1(314) 579-0066
Phoenix, AZ Office: +1(480) 752-0042

www.ociweb.com

OBJECT COMPUTING, INC.

Center of the System Administration

www.sbin.org

Open look at common things

Go Open

Open Source Software, which originally developed out of the passion of hackers and geeks, has now become the basis of real business solutions. Low TCO, high reliability, security and performance of Open Source software make the shift from "closed" technologies and standards to modern, open solutions inevitable.

CSA offers development, deployment and support services based on Open Source technologies and standards.
Up-to-date solutions for small and medium size businesses are available for you today with guaranteed full 24/7 support and no license fees.

Today, being closed means being left behind

+7 8793 365584

Designed by ITA "InfoMaster"

Free Software MAGAZINE

Contents

Issue 2, March 2005

EDITORIAL

Your part-time job 7

The trials and tribulations of being a “computer person”

FOCUS

The history and future of SMTP 10

by Kirk Strauser

SMTP’s adaptations to a hostile internet

Filtering spam with postfix 13

by Kirk Strauser

Effective ways to reduce unwelcome mail

Mail servers: resolving the identity crisis 18

by John Locke

How to get Dspam, Postfix, and Procmail to play well together

TECH WORLD

Poking at iTunes 27

by Chris J. Karr

A developer’s guide to the iTunes platform

Why free IT management tools are gaining traction 34

by Will Winkelstein

Enterprises are increasingly receptive to free software alternatives for IT management

Case study: Mythic Beasts 37

by Tony Mobily

A small company specialised in Linux servers and amazing support

Interview with Bernhard Reiter at aKademy 40

by Tom Chance

What we can do to promote the future of free software

Hard passwords made easy 43
by John Locke
Creating strong memorable passwords using mnemonic devices and word lists

WORD WORLD
The FUD-based Encyclopedia 49
by Aaron Krowne
Dismantling fear, uncertainty, and doubt, aimed at Wikipedia and other free knowledge resources

Guerrilla marketing 57
by Tom Chance
Part one: promoting community projects in the marketplace

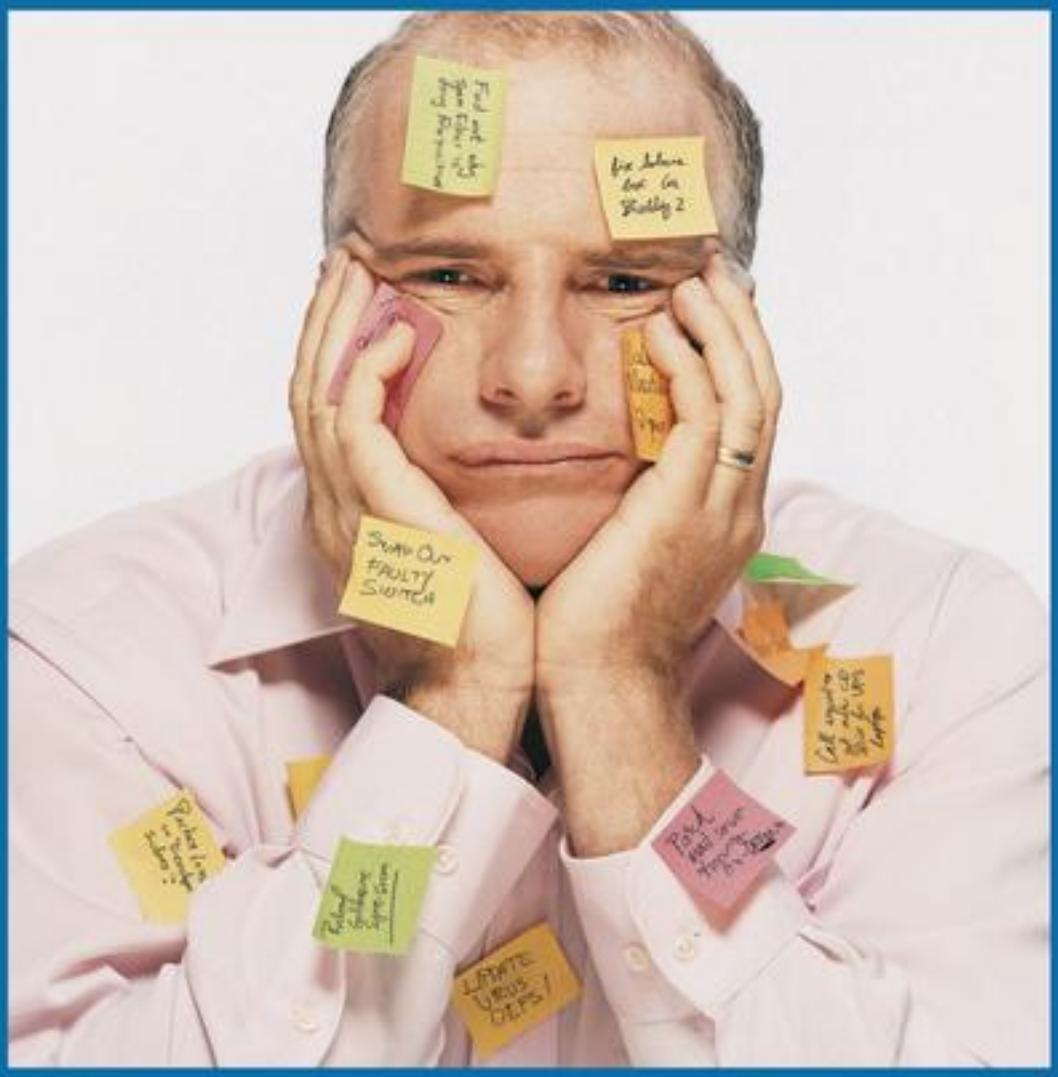
A law for free software 63
by Maureen O'Sullivan
Don't we have enough laws already?

The Libre Culture Manifesto 67
by David M. Berry, Giles Moss
A manifesto for free/libre culture

Richard Stallman's blog 73
by Richard Stallman
Selected entries from Richard's blog (http://agia.fsf.org/rms-blog), from November 2004 to December 2004

A PLANT NEEDS WATER TO GROW
Free Software
MAGAZINE
BY SUBSCRIBING YOU WILL BE SUPPORTING A MAGAZINE WHICH BELIEVES IN FREE SOFTWARE. ALL OUR ARTICLES ARE RELEASED UNDER THE GNU FREE DOCUMENTATION LICENSE, ENHANCING EXISTING INFORMATION ON FREE SOFTWARE.
SUBSCRIBE NOW!
WWW.FREESOFTWAREMAGAZINE.COM/SUBSCRIBE
FREE SOFTWARE MAGAZINE
NEEDS SUBSCRIBERS TO PROSPER!

Losing track of things?



There is a better way.

RT.

Enterprise-grade issue tracking.
Open source customizability.
Commercial support.

Download it now!

Best Practical Solutions offers training, support and custom development services. Check out our website or contact us for details.

www.bestpractical.com
sales@bestpractical.com

BEST PRACTICAL ™

Your part-time job

The trials and tribulations of being a "computer person"

Being a "computer person" these days is a very stressful business. Forget about angry customers, missed deadlines, unreasonable bosses and co-workers, shrinking wages, etc. Those are just things you get used to after a while.

For me, the real stress starts when I get out of the office, and I turn on my phone. I get the first call and I think: I wonder who that'll be; maybe Dave (yes, Dave Guard, our editor), maybe he wants to go and have a beer with me (he complained "But I don't drink beer?" when he edited this editorial...). But no, it's my cousin: his computer is behaving "oddly", and his internet connection has slowed down terribly (yes, he's sending spam... and he doesn't know it yet). He begs me to go and have a look at it. I try to explain to him that even though I am a computer person, I don't know anything about viruses on Windows XP. The begging goes on and on, until I give in and say with a sigh: all right, I'll come and have a look at it.

Andrea and I (he's a computer person too) worked out that, even for us, fixing a Windows XP machine takes about 8 hours, because we don't do it as a job and we're not fully equipped for it. To reinstall a system you need to back the previous information up, reinstall the system, install all the drivers (if the driver CDs ever existed), install all the patches and service packs, install the software that was previously installed, answer the question "Why is Office is not part of Windows?!?", discover that they had a crucial, weird device XYZ with no drivers available, and so on. This process is very stressful, and can go very wrong (in fact, I know it *never* goes smoothly). And after all of this you need to pray that doing, what you've spent hours doing, has solved the problem.

Now, I have a big family: 10 aunts and uncles, 25 cousins (including the ones once removed etc.), and roughly another 15 relatives who I'll put in the "other" category. That's 50 people, and on average, they'll have two "serious problems" with their computers every year. That's 100 "major incidents" (with all the begging and everything) per year.

A year is 365 days. 13 of them are public holidays. 104 are Saturdays and Sundays. 10 days for sick leave. So I'm left with 238 working days.

This means that if I were to respond to all of the emergency calls I receive, I would spend nearly half of my working life reinstalling Windows XP for my relatives. I obviously don't do that (otherwise, you wouldn't be reading this magazine). But saying "no" to your relatives at least twice a week is awkward.

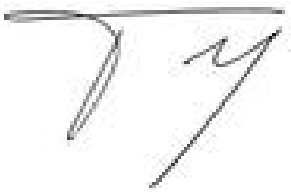
I'm writing this editorial after discovering that Microsoft is planning to buy Sybari Software, and will (probably) make more money off computer viruses. My own mother's computer is infected at the moment, and she is some 14000 Km away – I am powerless, and I can't even fix it by remote because the virus is using all her bandwidth. I feel angry, because I can't stop thinking that if Microsoft makes real money thanks to viruses, they'll never make their systems secure (and you know they could). And that really *is* a problem.

This month's issue is about spam, which is a problem closely related to viruses and Windows security. I'd have loved to publish an article entitled "How to solve the spam problem - forever", but I can't because such a solution doesn't exist yet.

I do look forward to the day when I can publish such an article, but until that day comes, I'll present you with some of the best ways we can defend ourselves against the invasion of spam.

Good luck...

Copyright information © by Tony Mobily
Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.



Free Software Magazine is a magazine by **The Open Company Partners Inc**, 90 Main St. Road Town, Tortola BVI

EDITOR IN CHIEF
Tony Mobily (t.mobily@)

TECHNICAL EDITORS
Clare James (c.james@)
Pancrazio De Mauro (p.demauro@)
Gianluca Insolvibile (g.insolvibile@)

EDITORS
Anna Dymitr Hawkes (a.dymitrhawkes@)
Dave Guard (d.guard@)

TECHS
Gianluca Pignalberi (LaTeX class and magazine generation) (g.pignalberi@)
Gian Maria Ricci (RTF to XML converter using VBA) (gm.ricci@)

GRAPHIC DESIGN
Alan Sprecacenere (Web, cover and advertising design) (a.sprecacenere@)
Tony Mobily, Gianluca Pignalberi, Alan Sprecacenere (Magazine design)

THIS PROJECT EXISTS THANKS TO
Donald E. Knuth, Leslie Lamport, People at TeX Users Group TUG (http://www.tug.org)

Every listed person is contactable by email. Please just add freesoftwaremagazine.com to the person's username in parentheses.

For copyright information about the contents of Free Software Magazine, please see the section "Copyright information" at the end of each article.

ALL MUSIC AND ARTWORK IS
RELEASED UNDER CREATIVE COMMONS
ATTRIBUTION SHARE-ALIKE LICENSE

WWW.LIBREISOCIETY.ORG
WWW.CREATIVECOMMONS.ORG
WWW.WIKIPEDIA.ORG

E-LOCA-TRONICA 2005
STICK TO WHAT YOU DON'T KNOW

HTTP://WWW.LOCARECORDS.COM

INFO@LOCARECORDS.COM

YOU CAN DOWNLOAD MUSIC,
VIDEOS AND TEXT BY VISITING
WWW.LOCARECORDS.COM

LOCA RECORDS, PO BOX 233,
BRIGHTON BN2 3WX, UK

DO ANDROIDS DREAM OF PROGRAMMING?

www.shop.software.com.pl

The history and future of SMTP

SMTP's adaptations to a hostile internet

Kirk Strauser

SMTP is an abbreviation for "Simple Mail Transfer Protocol", and is the standard internet protocol for sending email from one system to another. Although the word "simple" belies the inherent complexity of the protocol, SMTP has proved to be a remarkably robust, useful, and successful standard. The design decisions that made it so useful, though, have given spammers and infectious code an easy way to spread their unwanted messages. Its recent evolution reflects the tug-of-war between those unsavory players and the administrators who want to protect their systems and their users.

Early history

When Jonathan Postel wrote the SMTP definition RFC 821 in 1982, the internet was minuscule in comparison with today's pervasive mix of commercial, governmental, and private interests. At that time, it mostly comprised a small collection of military installations, universities, and corporate research laboratories. Connections were slow and unreliable, and the number of hosts was small enough that all of the participants could recognize each other. In this early setting, SMTP's emphasis on reliability instead of security was reasonable and contributed to its wide adoption. Most users helped each other by configuring their mail servers as "open relays". That meant that each cooperative host would accept mail meant for other systems and relay it toward its final destination. This way, email transfer on the fledgling internet stood a reasonable chance of eventual delivery. Most administrators were happy to help their peers – and receive their help in return.

Spam has existed since at least 1978, when an eager DEC sales representative sent an announcement of a product demonstration to a couple hundred recipients. The resulting outcry was sufficient to dissuade most users from repeating the experiment. This changed in the late 1990s: millions of individuals discovered the internet and signed up for inexpensive personal accounts and advertisers found a large and willing audience in this new medium.

Spam becomes a problem

The helpful nature of open relays was among the first victims of the spam influx. In the young commercial internet, high-speed connections were prohibitively expensive for individuals and small businesses. Spammers quickly learned that it was easy to send a small number of messages – with recipient lists thousands of entries long – to helpful corporate servers, which would happily relay those messages to their targets. Administrators noticed sudden spikes in their metered service bills (and in the number of complaints) and realized that they could no longer help their peers without incurring significant monetary costs and bad will.

First steps to secure the internet

Although the nature of the problem was clear, the solutions were not. The SMTP standard, which was designed with

reliability as a key feature, had to be re-implemented to purposefully discard certain, recognized messages. This was a foreign idea and no one was sure how to proceed.

The first step was to close the open relays. Administrators argued loudly, and at great length, whether this was a necessary move, or even a good one at all. In the end though, it was universally agreed that the trusting nature of the old internet was dead, and in fact harmful in the current setting.

Some users took this idea a step farther and decided that they would not only close their own systems, but would no longer accept messages from other open relays. They eventually began to share their lists of those relays with peers by adding specially formatted entries to their domain name servers and allowing their neighbors to query their servers for this data. This was the beginning of the first "DNS blackhole lists", and they were highly controversial. For example, administrators debated whether it was acceptable to actively test remote servers to see if they were open relays, and discussed which procedures a system administrator should follow to remove his or her host from the list after correcting the problem.

The first victims of "collateral damage" were those whose mail servers were blocked through no fault of their own. This often happened when over-zealous blacklist operators added entire blocks of addresses to their lists, rather than just the offending addresses. As one group of operators argued that the lists should err on the side of caution to prevent these problems, others believed that this would put extra pressure on the open relay administrators. In one form or another, this debate continues.

New threats

Huge numbers of people with very little computer-security experience came online, often with increasingly cheap, permanent high-speed connections. As a result, a new epidemic spread across the internet - most visibly as email worms. They infected poorly secured computers which then became the transmitters for new copies of those worms. Many of these propagate through the popular email clients on Microsoft Windows systems and move outward by emailing copies of themselves to people in the infected computer's address book. Many such infections are noticeable because they can overwhelm a machine and its internet connection to the point where both become useless to their end user, who then typically pay a business, or get a knowledgeable friend, to remove the worm.

There are more insidious infections which spread amongst computers rapidly. They then lie dormant to avoid drawing attention to themselves and wait for instructions from another system. A "botnet" is a collection of computers so compromised. Spammers often use botnets as a widely distributed means for sending large amounts of email.

Fighting back

A recent and popular response to these problems is sender authentication. That is, many mail servers now look for proof that a computer attempting to send email to them is actually authorized to do so. For example, Sender Policy Framework (or SPF) is centered around another specialized DNS record that lists the servers authorized to transmit email from a given domain. The administrator of example.com may list "smtp.example.com" and "mail.example.com" as the outbound mail servers for that domain. When an SPF-aware server receives a message from a user with an example.com email address, it compares the name of the computer attempting to send that message with those names. If it isn't on the list, then the message can reasonably be assumed to be a forgery and may be discarded. Several proposals exist that are similar to SPF, such as Yahoo!'s DomainKeys, but all work in essentially the same way.

Another common measure is simply to enforce the SMTP definition and reject messages that do not adhere to it. This is highly effective because few, if any, worms or spam transmitters bother to comply with the standards. They often take shortcuts when generating the email address that a message claims to originate from, or lie about their own identity. Some seem to completely ignore the standard in hopes that the receiving system will blindly process their load anyway. The methods of enforcing the protocol must be implemented incrementally, though, as many old but legitimate mail servers may also fail to meet some of the more pedantic requirements. An old rule of networking is to "be liberal in what you accept". Sadly, spammers seem to be on the brink of making that impossible.

One of the positive side effects of sender authentication and standards enforcement is that email senders are being compelled to correctly identify themselves before they are al-

lowed to transfer their messages. New DNS blackhole lists, able to narrowly identify specific senders, will be possible once a critical mass of servers have implemented such measures. This solution should neatly avoid the old problem of collateral damage, as well as greatly reducing the scope of the blackhole lists themselves.

Regardless, some spam and worms will always make it through the tightest of filters. To this end, a new class of intelligent, self-learning filtering software does a good job of identifying the remaining unwanted messages. Good, free antivirus programs also perform well at removing worm-infected messages before they can reach vulnerable email clients.

One of the newer and more exotic approaches is known as greylisting. The idea is simple: receiving mail servers make senders wait for a small amount of time before they are allowed to transmit email to a recipient they've never sent to before. This serves two purposes. First, very few worms, spam senders, or botnet machines actually have the patience to try again later (or the resources to remember which addresses should be retried). If their first attempt at delivering a message fails, they give up and move on to the next destination. Second, by increasing the effective length of time it takes for a spammer to send a message, a mail server also increases the chances that a DNS blackhole list will add that rogue server before it can deliver that message. Few methods can compete with the simple elegance of greylisting, and it offers to many frustrated administrators the hope that the war against unwanted email can be won.

Finally, some administrators have responded to the overwhelming loads which are sometimes sent by botnets by blocking certain operating systems. Almost no one runs a legitimate mail server on Window 98, for example. Therefore, configuring a firewall to block incoming SMTP connections from Windows 98 machines (assuming all of your desktop clients use newer version of Windows, or Mac or Unix desktops) can reduce the number of unwanted messages from hijacked computers.

The future

SMTP has a long and illustrious past. It's one of the "killer applications" that led to the explosive growth of the internet. From love letters to stock transactions to family photos, countless users send an endless variety of messages to each other every day. Email in its current form is going to be around for a long time, but will likely undergo a series of incremental updates. For example, client authentication (which didn't exist when the SMTP RFC was written) has almost completely replaced open relaying, and some mail servers now use SSL certificates to verify another server's identity.

However, the future of SMTP depends largely upon those who abuse it. It currently provides a reliable, fault-tolerant system of email delivery. Any changes are likely to work against this reputation, as they would add to the complexity of the protocol. Several proposed alternatives have come and gone, and there are no widely accepted proposals that stand a reasonable chance of coming into common use. Only time will tell…

Bibliography

[1] RFC 821 – Simple Mail Transfer Protocol (http://www.faqs.org/rfcs/rfc821.html)

[2] Reaction to the DEC spam of 1978 (http://www.templetons.com/brad/spamreact.html)

[3] Wikipedia entry on DNSBLs (http://en.wikipedia.org/wiki/DNSBL)

[4] Sender Policy Framework (http://spf.pobox.com/)

[5] DomainKeys: Proving and Protecting Email Sender Identity (http://antispam.yahoo.com/domainkeys)

Copyright information

© 2005 by Kirk Strauser

This article is made available under the "Attribution-Share-alike" Creative Commons License 2.0 available from http://creativecommons.org/licenses/by-sa/2.0/.

About the author

Kirk Strauser has a BSc in Computer Science from Southwest Missouri State University. He works as a network application developer for The Day Companies, and runs a small consulting firm (http://www.strausergroup.com/) that specializes in network monitoring and email filtering for a wide array of clients. He has released several programs under free software licenses, and is active on several free software support mailing lists and community websites.

Filtering spam with postfix

Effective ways to reduce unwelcome mail

Kirk Strauser

If you are responsible for maintaining an internet-connected mail-server, then you have, no doubt, come to hate spam and the waste of resources which comes with it. When I first decided to lock down my own mail-server, I found many resources that helped in dealing with these unwanted messages. Each of them contained a trick or two, however very few of them were presented in the context of running a real server, and none of them demonstrated an entire filtering framework. In the end I created my own set of rules from the bits and pieces I found, and months of experimentation and fine-tuning resulted in the system detailed in this article.

This article will show you how to configure a Postfix mail-server in order to reject the wide majority of unwanted incoming "junk email", whether they contain unsolicited commercial email (UCE), viruses, or worms. Although my examples are specific to Postfix, the concepts are generic and can be applied to any system that can be configured at this level of detail.

For a real world example, I'll use my server's configuration details:

Tab. 1: My server's configuration details

Hostname	Kanga.honeypot.net
Public address	208.162.254.122
Postfix configuration path	/usr/local/etc/postfix

It can be difficult to find a good compromise between safety and effectiveness, but I believe I've found a solid combination that should work in almost any situation.

Goals

This configuration was written with two primary rules in mind:

1. Safety is important above all else. That is, I would much rather incorrectly allow an unwanted message through my system than reject a legitimate message.
2. The system had to scale well. A perfect system that uses all of my server's processing power at moderate workloads is not useful.

To these ends, several checks - that may have reduced the number of "false negatives" (messages that should have been rejected but were not) at the cost of increasing the number of "false positives" (legitimate messages that were incorrectly rejected) - were avoided. The more resource-intensive tests were moved toward the end of the configuration. This way, the quick checks at the beginning eliminated the majority of UCE so that the longer tests had a relatively small amount of traffic to examine.

HELO restrictions

When a client system connects to a mail-server, it's required to identity itself using the `SMTP HELO` command. Many viruses and spammers skip this step altogether, either by sending impossibly invalid information, or lying and saying that they are one of your own trusted systems - in the hopes that they will be granted undeserved access. The first step in the filtering pipeline, then, is to reject connections from clients that are severely mis-configured or that are deliberately attempting to circumvent your security. Given their simplicity, these tests are far more effective than might be imagined, and they were implemented in my `main.cf` file with this settings block:

```
1  smtpd_delay_reject = yes
2  smtpd_helo_required = yes
3  smtpd_helo_restrictions =
4     permit_mynetworks,
5     check_helo_access
           hash:/usr/local/etc/postfix/helo_access,
6     reject_non_fqdn_hostname,
7     reject_invalid_hostname,
8     permit
```

Line 1 is a fix for certain broken (but popular) clients, and is required in able to use `HELO` filtering at all. The second line rejects mail from any system that fails to identify itself. Line 4 tells Postfix to accept connections from any machines on my local network. The next line references an external hash table that contains a set of black- and whitelisted entries; mine looks like this:

```
woozle.honeypot.net     OK
honeypot.net            REJECT You are not me. Shoo!
208.162.254.122         REJECT You are not me. Shoo!
```

The first line in the table explicitly allows connections from my laptop so that I can send mail when connected through an external network. At this point Postfix has already been told to accept connections from all of my local machines and my short list of remote machines, so any other computer in the world that identifies itself as one of my systems is lying. Since I'm not particularly interested in mail from deceptive systems, those connections were flatly rejected.

Lines 6 through 8 complete this stage by rejecting connections from hosts that identify themselves in blatantly incorrect ways, such as "MAILSERVER" and "*HOST@192.168*!aol.com".

Some administrators also use the `reject_unknown_hostname` option to ignore servers whose hostnames can't be resolved, but in my experience this causes too many false positives from legitimate systems with transient DNS problems or other harmless issues.

You can test the effect of these rules, without activating them on a live system, by using the `warn_if_reject` option to cause Postfix to send debugging information to your maillog without actually processing them. For example, line 6 could be replaced with:

```
warn_if_reject,
reject_non_fqdn_hostname,
```

This way the results can be observed without the risk of inadvertently getting false positives.

Sender restrictions

The next step is to reject invalid senders with these options:

```
9  smtpd_sender_restrictions =
10    permit_sasl_authenticated,
11    permit_mynetworks,
12    reject_non_fqdn_sender,
13    reject_unknown_sender_domain,
14    permit
```

Given their simplicity, these tests are far more effective than might be imagined

Lines 10 and 11 allow clients that have authenticated with a username and password or Kerberos ticket, or who are hosts on my local network, to continue onward. Lines 12 and 13 work similarly lines 6 and 7 in the "HELO restrictions" section; if the sender's email address is malformed or provably nonexistent, then there's no reason to accept mail from them. The next line allows every other message to move on to the next phase of filtering.

Recipient restrictions and expensive tests

By this time, it's clear that the client machine isn't completely mis-configured and that the sender stands a reasonable chance of being legitimate. The final step is to see that

the client has permission to send to the given recipient and to apply the remaining "expensive" tests to the small number of messages that have made it this far. Here's how I do it:

```
15 smtpd_recipient_restrictions =
16   reject_unauth_pipelining,
17   reject_non_fqdn_recipient,
18   reject_unknown_recipient_domain,
19   permit_mynetworks,
20   permit_sasl_authenticated,
21   reject_unauth_destination,
22   check_sender_access
       hash:/usr/local/etc/postfix/sender_access,
23   check_recipient_access
       hash:/usr/local/etc/postfix/recipient_access,
24   check_helo_access
       hash:/usr/local/etc/postfix/secondary_mx_access,
25   reject_rbl_client relays.ordb.org,
26   reject_rbl_client list.dsbl.org,
27   reject_rbl_client sbl-xbl.spamhaus.org,
28   check_policy_service unix:private/spfpolicy
29   check_policy_service inet:127.0.0.1:10023
30   permit
```

Many spammers send a series of commands without waiting for authorization, in order to deliver their messages as quickly as possible. Line 16 rejects messages from those attempting to do this.

Options like lines 17 and 18 are probably becoming familiar now, and they work in this case by rejecting mail targeted at domains that don't exist (or can't exist). Just as in the "Sender restrictions" section, lines 19 and 20 allow local or authenticated users to proceed – which here means that their messages will not go through any more checks. Line 21 is critically important because it tells Postfix not to accept messages with recipients at domains not hosted locally or that we serve as a backup mail server for; without this line, the server would be an open relay!

The next line defines an access file named sender_access that can be used as a black- or whitelist. I use this to list my consulting clients' mail-servers so that none of the remaining tests can inadvertently drop important requests from them. I added line 23, which creates a similar blacklist called recipient_access for recipient addresses, as an emergency response to a "joe job". Once in 2003, a spammer forged an email address from my domain onto several million UCE messages and I was getting deluged with bounce messages to "michelle@honeypot.net". I was able to reject these by adding an entry like:

```
michelle@honeypot.net REJECT This is a forged account.
```

Although the event was annoying, this allowed my server to continue normal operation until the storm had passed.

Line 24 is the last of the "inexpensive" checks. It compares the name that the remote system sent earlier via the `HELO` command to the list of my secondary mail servers and permits mail filtered through those systems to be delivered without further testing. This is the weak link in my filtering system, because if a spammer were clever enough to claim that they were one of my backup servers then my mail server would cheerfully deliver any message sent to it. In practice, though, I've never seen a spammer that crafty and this line could be removed without side effects should the need arise.

Lines 25 through 27 are somewhat more controversial than most of my techniques, in that they consult external DNS blackhole lists in order to selectively reject messages based on the IP address of the sender. Each of these lists have been chosen because of their good reputation and very conservative policies toward adding new entries, but you should evaluate each list for yourself before using their databases to drop messages.

SPF and greylisting

Lines 28 and 29 add Sender Policy Framework (SPF) and greylisting filtering respectively. SPF works by attempting to look up a DNS record, that domains can publish, which gives the list of addresses allowed to send email for that domain. For example, webtv.net's SPF record is currently "v=spf1 ip4:209.240.192.0/19 -all", which means that a message claiming to be from joeuser@webtv.net sent from the IP address 64.4.32.7 is forged and can be safely rejected.

Greylists are pure gold when it comes to rejecting junk email. Whenever a client attempts to send mail to a particular recipient, the greylist server will attempt to find that client's address and the recipient's address in its database. If there is no such entry then one will be created, and Postfix will use a standard SMTP error message to tell the client that the recipient's mailbox is temporarily unavailable and to try again later. It will then continue to reject similar attempts until the timestamp is of a certain age (mine is set to five

minutes). The theory behind this is that almost no special-purpose spam sending software will actually attempt to re-send the message, but almost every legitimate mail server in existence will gladly comply and send the queued message a short time later. This simple addition cut my incoming junk email load by over 99% at the small cost of having to wait an extra five minutes to receive email for the first time from a new contact. It has worked flawlessly with the many mailing lists that my clients and I subscribe to and has not caused any collateral problems that I am aware of. If you take nothing else from this article, let it be that greylisting is a *Good Thing* and your customers will love you for using it.

I use the smtpd-policy.pl script that ships with Postfix to handle SPF, and Postgrey as an add-on greylisting policy server. They're defined in my master.cf file as:

```
spfpolicy unix -    n    n    -    -    spawn
   user=nobody argv=/usr/bin/perl
         /usr/local/libexec/postfix/smtpd-policy.pl
greypolicy unix -    n    n    -    -    spawn
   user=nobody argv=/usr/bin/perl
         /usr/local/libexec/postfix/greylist.pl
```

Content filtering

The messages remaining at this point are very likely to be legitimate, although all that Postfix has actually done so far is enforce SMTP rules and reject known spammers. Their final hurdle on the way from their senders to my users' mailboxes is to pass through a spam filter and an antivirus program. The easiest way to do this with Postfix is to install AMaViS, SpamAssasin, and ClamAV and then configure Postfix to send messages to AMaViS (which acts as a wrapper around the other two) before delivering them, and line 31 does exactly that:

```
31 content_filter = smtp-amavis:127.0.0.1:10024
```

SpamAssassin is fantastic at identifying spam correctly. Its "hit rate" while used in this system will be much lower than if it were receiving an unfiltered stream, as most of the easy targets have already been removed. I recommend setting AMaViS to reject only the messages with the highest spam scores; I arbitrarily picked a score of 10.0 on my systems. Then, tag the rest with headers that your users can use to sort mail into junk folders.

ClamAV has proven itself to be an effective and trustworthy antivirus filter, and I now discard any message that it identifies as having a viral payload.

Unfortunately, the configuration of these systems is more complicated than I could hope to cover in this article, as it depends heavily on the setup of the rest of your email system. The good news is that literally less than 1% of junk email makes it this far into my system, and I've actually gone for several days without realizing that SpamAssassin or ClamAV hadn't been restarted after an upgrade. Still, these extra filters are very good at catching those last few messages that make it through the cracks.

Other possibilities

If you want more aggressive filtering and can accept the increased risk of false positives, consider some of the other less-conservative blackhole lists such as the ones run by SPEWS (`http://www.spews.org`) or the various lists of blocks of dynamic IP addresses. You may also consider using the reject_unknown_hostname option mentioned in the "HELO restrictions" section, but you can expect a small, measurable increase in false positives.

The ruleset described above should be sufficient on its own to eliminate the vast majority of junk email, so your time would probably be better spent implementing and adjusting it before testing other measures.

Conclusion

None of the techniques I use are particularly difficult to implement, but I faced quite a challenge in assembling them into a coherent system from the scraps I found laying about in various web pages and mailing lists.

The most important thing I learned from the process was that it's easy to experiment with Postfix, and it can be customized to your level of comfort. When used in my configuration, the most effective filters are:

- Greylisting
- DNS blackhole lists
- HELO enforcement

Greylisting has proven to be an excellent filter and I've deployed it successfully on several networks with nothing

but positive feedback from their owners. Even the basic HELO filtering, though, can visibly decrease spam loads and should be completely safe. It can be difficult to find a good compromise between safety and effectiveness, but I believe I've found a solid combination that should work in almost any situation. Don't be afraid to test these ideas on your own and make them a part of your own anti-spam system!

Bibliography

[1] Postfix Configuration - UCE Controls (http://www.postfix.org/uce.html)

[2] SPF: Sender Policy Framework (http://spf.pobox.com/)

[3] Greylisting.org - a great weapon against spammers (http://greylisting.org/)

[4] Postgrey - Postfix Greylisting Policy Server (http://isg.ee.ethz.ch/tools/postgrey/)

[5] AMaViS - A Mail Virus Scanner (http://www.amavis.org/)

[6] The Apache SpamAssassin Project (http://spamassassin.apache.org)

[7] Clam AntiVirus (http://www.clamav.net)

Copyright information

© 2005 by Kirk Strauser

This article is made available under the "Attribution-Share-alike" Creative Commons License 2.0 available from http://creativecommons.org/licenses/by-sa/2.0/.

About the author

Kirk Strauser has a BSc in Computer Science from Southwest Missouri State University. He works as a network application developer for The Day Companies, and runs a small consulting firm (http://www.strausergroup.com/) that specializes in network monitoring and email filtering for a wide array of clients. He has released several programs under free software licenses, and is active on several free software support mailing lists and community websites.

Linux Based Solutions
www.snowdoniait.com
Tel: 01248 355 985 Mob: 07976 632 572
SNOWDONIA
IT SERVICES
By using our services you will gain the many benefits of Linux and free software:
• A secure, stable operating system
• A reduction in licensing costs
• A wide variety of applications for home and office use
• A powerful, customisable system
Services available include:
• Installation and configuration of server and desktop systems with appropriate applications
• Training in the use of your Linux system
• Website design with hosting on Linux servers
• Monitoring systems with customised plugins
• Vulnerability scanning and information security programme development
Contact us for a demonstration of Linux and the applications available, or to enquire about our Linux based solutions for home users and businesses.
On-site daytime and evening visits available. Competitive rates, free estimates.

Mail servers: resolving the identity crisis

How to get Dspam, Postfix, and Procmail to play well together

John Locke

Dspam (http://nuclearelephant.com/projects/dspam) filters spam with the best. In my installation, it stops over 98% of all spam: I've only had one false positive in the last year, and that was a message to the Dspam list that contained a real spam!

Administering Dspam is a breeze. No rules to configure, new users can automatically benefit from a global dictionary and quarantine management is simple. But getting a Dspam quarantine set up the first time, without losing any email, can challenge the most seasoned mail administrators.

In this article, I'm going to trace email through the various parts of a Postfix mail server running Dspam and Procmail or Maildrop, focusing on getting the CGI quarantine working correctly. I'll provide the configuration steps for local users. I'll also explain where the configuration differs for virtual users, and the basic changes or decisions that you need to consider, but I won't provide the actual configuration for virtual users in this article.

The ingredients

There are many different recipes for putting together a mail server on Unix. I'm going to use these programs because they're as good, or better than, anything else available.

First, install all of this software according to your distribution. In most cases, you should be able to install Postfix, Apache, mod_suexec, and Procmail through your distribution's package management system. Dspam and Maildrop require custom configuration at compile time, so they usually need to be compiled from source code.

Tab. 1: Programs discussed in this article

Program	Role	User IDs
Postfix	Mail Transfer Agent (MTA)	Root Postfix
Dspam	Spam filter	Actual User ID mail dspamcgi
Apache with mod_suexec	Quarantine	Dspamcgi apache/nobody
Procmail Maildrop	Local Delivery Agent (LDA)	Root Actual User ID Virtual User ID

Getting Postfix running

Administering Postfix, or any other MTA, is the subject of many a book and it's hard to do justice to the topic in a few hundred words. Before even attempting to get Dspam working, you should have Postfix up and running and successfully delivering to your mailboxes. You should have Postfix rejecting mail for invalid email accounts — the days when a "catchall" account caught anything other than spam are long gone. Besides, it's much better to have real email

users learn immediately when they've sent something to the wrong address, than for their message to get through and sit in quarantine with a few thousand spams.

Postfix is actually a collection of smaller programs, each dedicated to a particular task. One program talks to the outside world. Another cleans up message headers and determines the next place to direct a message. Others check that a message is valid.

Postfix comes with two different Local Delivery Agents (LDAs). These are the programs that actually deliver mail to a mailbox. The two LDAs are called Local and Virtual(8).

Local delivers mail to real Unix user accounts. When a remote mail server connects and attempts to send an email, Local can verify whether the corresponding local user account exists, allowing Postfix to accept or reject the message before even receiving it.

Virtual(8) can deliver mail to a set of virtual users (users who don't actually have a local Unix account); in the Postfix main configuration file, Virtual(8) uses directives starting with `virtual_mailbox`. There is also a Virtual(5), which is used to rewrite the destination address of an email before attempting delivery — this is entirely different, using `virtual_alias` directives in the Postfix configuration files. The numbers refer to the section of the man pages containing the documentation for the program. Virtual(8) uses a specified user id to deliver mail to any virtual user, and is limited to a single directory structure that you specify. Both Virtual(5) and Virtual(8) can be used to reject unknown users, exactly the same as Local.

Read the man pages to learn about these programs:

```
# man 8 local
# man 8 virtual
# man 5 virtual
```

Before adding any other programs to your mail server, get Postfix delivering mail to the correct place using one of its built-in LDAs. There are many How-Tos on the web to help you get this done — start with the links available at the Postfix (`http://www.postfix.org`) web site.

There are many different recipes for putting together a mail server on Unix. I'm going to use these programs because they're as good, or better than, anything else available

What happens when an email arrives

Postfix runs a process called Master, to start other processes as necessary. Master runs as root, and changes user ID, as appropriate, when starting its other processes. A process called Smtpd listens to port 25, running as the postfix user with very few permissions. Smtpd handles all communication with the remote mail software.

Before accepting any mail, Postfix runs through a series of checks. You can have it check the remote address, the validity of the server greeting, the sender address, and the recipient address for all kinds of different criteria (see the article by Kirk Strauser (`http://www.freesoftwaremagazine.com/free_issues/issue_02/focus_spam_postfix/`) in this magazine for more detailed information on Postfix). At a minimum, use the recipient checks to only allow mail going to or from known users — you don't want your server to become an *Open Relay*, a mail server that allows unknown people to send mail to anyone. Here's a reasonable set of recipient checks:

```
smtpd_recipient_restrictions =
  reject_non_fqdn_sender,
  reject_unknown_recipient_domain,
  reject_unauth_pipelining,
  reject_non_fqdn_recipient,
  permit_mynetworks,
  permit_sasl_authenticated,
  reject_unauth_destination,
  permit
```

In this example, the `permit_mynetworks` and `permit_sasl_authenticated` statements allow known users to send valid email anywhere. (Setting up SASL authentication is a topic for another article...) The key blocking directive here is `reject_unauth_destination`. This directive tells Postfix to see if it knows exactly where to deliver the message. It will check Local, Virtual(8), Virtual(5), its list

of relay domains, and a few other places to see if it can deliver the mail to a known location. If not, Smtpd returns a REJECT code, without ever accepting the message to be delivered.

Otherwise, Smtpd receives the message and hands it off to another internal Postfix process. After going through a number of processes, which you can configure for content filtering and other tasks, the incoming mail eventually reaches the point where it's going to be delivered to a mailbox. The recipient address is rewritten according to alias and virtual alias rules, and matched to a transport. The transport is used to determine which LDA to use: Local, Virtual(8), or another transport defined in the `/etc/postfix/master.cf` file. For a local Unix user, the mail is handed to the Local transport, started as the actual Unix user id. Local then uses the command specified in the `mailbox_command` directive to deliver the message to a mailbox.

Inserting Dspam or another LDA

There are basically three different places where you can insert Dspam into the Postfix mail delivery sequence:

1. As a content filter, before delivery.
2. As a transport.
3. As the mailbox command for the Local LDA.

It's theoretically possible to use the first option and insert Dspam as a content filter before even resolving the destination address. If you're deploying it on a relay server, this is a good option, but this goes beyond the scope of the article. Setting up Dspam this way gets pretty tricky, but it's possible to integrate with Amavisd-new along with an antivirus engine, configuring all users to share a single token dictionary.

The main issue with the other two options is that once Postfix delivers the message to Dspam, Dspam can't get the message back into Postfix for actual delivery. You can't use either of Postfix's native LDAs, Local or Virtual(8). This means that before deploying Dspam, you must get another LDA installed and working.

The same thing is true when you configure the Dspam quarantine — Dspam needs an LDA to deliver false positive messages (messages flagged as spam that are really innocent mail). If you use Dspam as a content filter, it's difficult to set up the quarantine — much easier to configure Dspam to add its header, deliver spam and innocent mail, and separate the spam at a later stage.

Fortunately, Procmail is a perfectly fine LDA included with most Linux distributions, and Maildrop is pretty easy to obtain and install. Which should you use? If you only have local users, Procmail is probably already on your machine and probably needs no further configuration. If you're trying to support virtual users, Maildrop is much easier to set up because it can look up virtual users from a number of different sources.

LDA user ids and permissions

If you understand Unix permissions, you probably know that only programs running as the root user can change user ids. When you start a program, it uses your user id to execute — unless it has a special bit set, the SetUID or SetGID bit. Programs that are SetUID and owned by the root user will run as the root user, regardless of which user account was used to start the program. Obviously, this can be a huge security risk, because if the program isn't carefully written to protect exploits, any user on the system could potentially gain root access.

Procmail is a perfectly fine LDA included with most Linux distributions, and Maildrop is pretty easy to obtain and install

When you use the Postfix Local LDA to deliver to a command, the command is started with the user id of the recipient. This works great for delivering mail, and adding Dspam to the mix doesn't cause any further problems with initial delivery. You can set Procmail as the delivery command by adding this directive to `/etc/postfix/main.cf`:

```
mailbox_command = /usr/bin/procmail
```

Naturally, if Procmail is in a different location, change the path as necessary.

The problem comes when you add delivery from the quarantine, or if you use the transport method to deliver to Dspam.

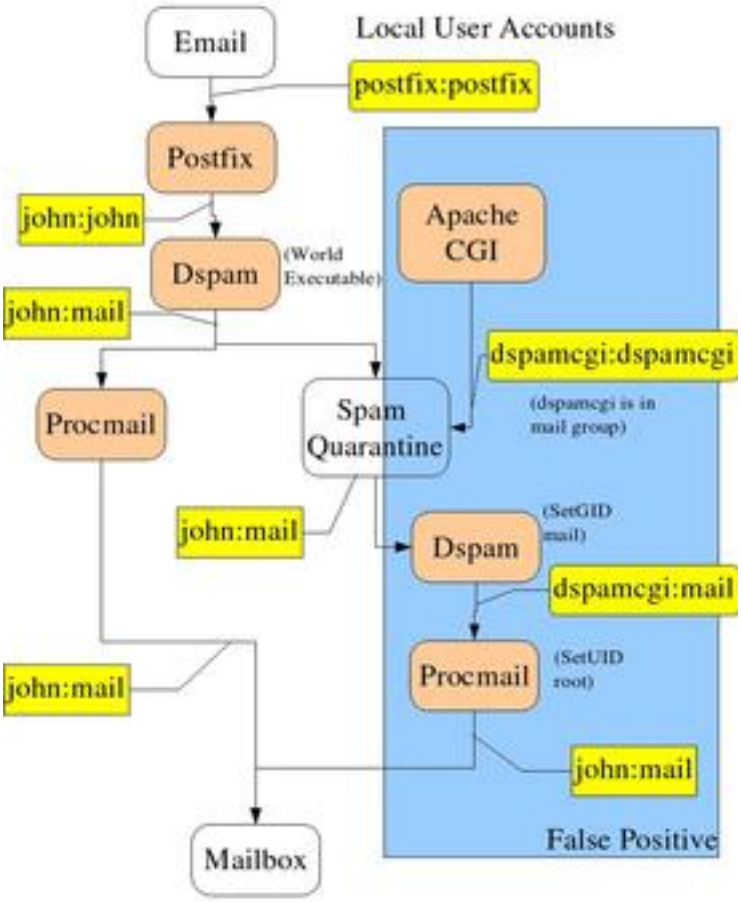


Fig. 1: To get Dspam and Procmail working correctly together, trace the execution of the various programs handling each email through the entire delivery process. Pay special attention to the effective user and group ids of each program.

In either case, Dspam will run under a different user account, *not* the user account of the user receiving the mail. At some point, the LDA needs to change to the correct user account to be able to deliver the mail. For it to be able to do this, it needs to be SetUID root.

Procmail delivery modes

Because Procmail often needs to be SetUID root, it limits the amount of time it spends as root, changing to the receiving user id as soon as it can. Postfix can call Procmail in either of two of its delivery modes: *Deliver* mode, or *Normal* mode.

In Normal mode, Procmail is called without any arguments, and delivers to the mailbox for the user id which is used to call Procmail. As shown, this works for normal delivery, even with Dspam. However, you cannot easily get Dspam to change to the real user account, when it's called by the quarantine CGI, so if you're going to use the quarantine, this mode won't meet your needs.

In Delivery mode, Procmail must be launched with the `-d` switch and a real user ID. For example:

```
/usr/bin/procmail -d john
```

For this to work, Procmail must either be SetUID root, or started by the root user. You can find out if Procmail is SetUID root by listing the Procmail binary, which should look like one of these:

```
# ls -l /usr/bin/procmail
-rwsr-sr-x 1 root mail /usr/bin/procmail
```

or

```
-rwsr-xr-x 1 root mail /usr/bin/procmail
```

To set Procmail to be SetUID root, do these commands:

```
# chown root /usr/bin/procmail
# chmod 4755 /usr/bin/procmail
```

Procmail allows any user account to use delivery mode, but requires that the destination user specified be a valid Unix user account.

Maildrop Trusted Users

If you choose to use Maildrop instead of Procmail, you still need to set it to SetUID root. You also need to specify at compile time which users are allowed to use its equivalent of delivery mode — Maildrop returns an error if an untrusted user tries to deliver to a different user account. The advantage of Maildrop is that you can specify a virtual user account to deliver to, and Maildrop will look up, in any one of several different locations, to determine where to deliver the message.

You can download Maildrop from here (`http://www.courier-mta.org/maildrop`).

Here's what my configure line for Maildrop looks like:

```
$ ./configure --enable-syslog=1 --enable-maildropmysql \
  --enable-trusted-users="root dspamcgi mail"
```

The crucial step here is the `--enable-trusted-users` flag. In my case, I've allowed three users to utilize Maildrop to deliver to other users: root, dspamcgi, and mail. I've specified the root user to make troubleshooting easier. Dspamcgi is a user account I created for the CGI quarantine — you'll need to create this user account before specifying it in the Maildrop configure line. Mail, in my case, is the Unix user account that I use to have Postfix deliver as a transport. I'll get to this in a little bit.

After the usual `./configure`, just run `make` and `make install`, and Maildrop should be ready to go. It should look like this:

```
$ ls -l /usr/local/bin/maildrop
-rwsr-xr-x 1 root mail 165616 Oct 4 07:45 /usr/local/bin/ma
```

You can configure Postfix to call Maildrop as a transport, or use the same `mailbox_command` directive as I used for Procmail:

```
mailbox_command = /usr/local/bin/maildrop
```

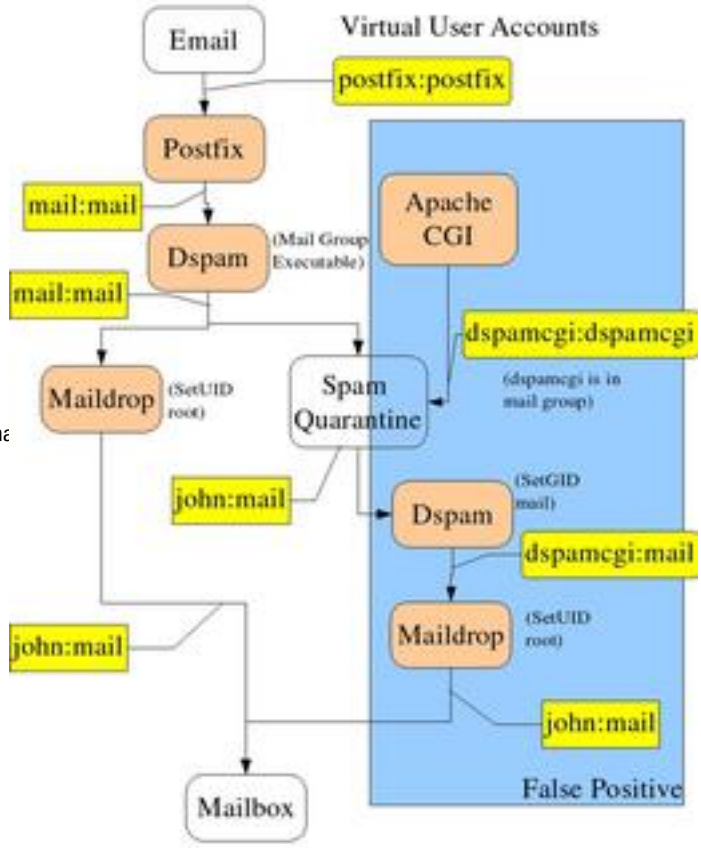
Adding Dspam to the chain

Before adding Dspam, get either Maildrop or Procmail working correctly. Verify that mail gets delivered to the right mailbox, and that invalid email addresses are rejected rather than bounced. Once you have an external LDA working correctly, adding Dspam is pretty simple. Download the most recent stable version from here (`http://nuclearelephant.com/projects/dspam`), and unpack the tarball.

You used to need to specify a delivery agent on the configure line, and make assorted setting in different files, but now most options are in the `dspam.conf` file.

Compile and install Dspam using the instructions in the README. The main decision you'll have to make is the storage driver. The MySQL driver is the most stable and best performing driver — if you have installed MySQL from your distribution sources, you'll also need to install the development packages.

Fig. 2: Maildrop can be configured exactly like Procmail. However, Maildrop can also deliver to virtual user accounts, mail accounts that do not exist as local Unix users. This diagram traces the flow of mail and ownership of processes when delivering to a virtual email account.



The README contains instructions for configuring Dspam as a transport in Postfix. To configure as a mailbox command for the Postfix Local LDA, read on.

When Postfix delivers to Dspam as a transport, you need to specify a particular Unix user account that Postfix can use to start Dspam. In my case, I use the Mail user. Whatever user you list here must be added to `dspam.conf` as a trusted user.

On the other hand, if Dspam is called from the Local LDA, it runs as a real Unix user, the user to be used for delivery. In this case, do not add individual users as trusted users `in dspam.conf` — they are not trusted users. Dspam will run under the specified user account, and use the `UntrustedDeliveryAgent` specified in

`dspam.conf` to deliver the message.

Unlike the LDA, Dspam does not need to be SetUID when used with Postfix. But if you're using the CGI quarantine, it does need to be SetGID.

> The advantage of Maildrop is that you can specify a virtual user account to deliver to, and Maildrop will look up, in any one of several different locations, to determine where to deliver the message

To get Dspam to process mail and hand off to Procmail as a mailbox_command (option #3), you need to make three changes:

- Change the permissions on the Dspam binary to world executable:

```
# chmod o+x /usr/local/bin/dspam
```

- Change the delivery agents in `Dspam.conf`:

```
UntrustedDeliveryAgent "/usr/bin/procmail"
TrustedDeliveryAgent "/usr/bin/procmail -d %u"
```

- Change the Postfix `mailbox_command` in `/etc/postfix/main.cf` to deliver to Dspam:

```
mailbox_command = /usr/local/bin/dspam
   --deliver=innocent --user $USER
```

To get Dspam to process mail as a transport (option #2) is a bit trickier. It involves setting up a Dspam transport in `/etc/postfix/master.cf`, adding a transport map that tells Postfix which addresses to deliver to the Dspam transport, and setting the `destination_recipient_limit` for the Dspam transport to 1.

Dspam Binary Permissions

The CGI quarantine for Dspam runs under the web server. Dspam creates a bunch of data files under its `$DSPAM_HOME`, by default `/var/lib/dspam`. Because Dspam can run under several different user accounts, and Dspam needs to write to many of these files as different users, Dspam sets all of these files as group writeable. To make this work, Dspam must always run under the same group ID, regardless of its user id. By default, Dspam installs itself SetGID to the mail group, so it should work for you correctly.

As I noted above, if you use the `mailbox_command` method of launching Dspam, untrusted users will be calling Dspam, so Dspam needs to be world executable. If you use transport mode, Dspam will always be called by the user you specify in the transport, in `/etc/postfix/master.cf`, so Dspam doesn't need to be world executable. But in this case, you need to add all of the user accounts that call Dspam to whatever group Dspam is set to use.

Finally, if you use Procmail and the CGI quarantine, either Dspam or Procmail needs to be SetUID root.

Training Aliases

Dspam comes with a handy tool to generate spam training aliases, called `dspam_genaliases`. It parses `/etc/passwd`, finds all valid users, and outputs a list of users prefixed with "spam-". You can write this output straight to a file and set Postfix to use it. To get it to work for all users, do something like this:

```
# dspam_genaliases --minuid 500 >
    /etc/postfix/dspam.aliases
# postalias /etc/postfix/dspam.aliases
```

Then add the file to the Postfix `alias_maps` directive in `/etc/postfix/main.cf`:

```
alias_maps = hash:/etc/aliases hash:
             /etc/postfix/dspam.aliases
```

Remember, you'll need to run this whenever you add or delete a user from the system.

This method becomes much more cumbersome if you support virtual users. You can use a regular expression transport map and a training transport in Postfix to avoid having to generate an alias file for virtual users, but you can't easily have the "spam" part added to the front of the address. Search the Dspam mailing list (via Google) to find out how.

Tab. 2: Dspam Binary Permissions

Postfix configuration	Local Delivery Agent	Dspam permissions	Command to set permissions
mailbox_command	Procmail, SetUID Root Maildrop, SetUID Root	-r-x--s--x root mail	# chmod 2511
mailbox_command	Procmail, not SetUID Root	-r-s--s--x root mail	# chmod 6511
Transport	Procmail, SetUID Root Maildrop, SetUID Root	-r-x--s--- root mail (all delivery users must be added to Unix mail group and Dspam Trusted Users)	# chmod 2510

Apache User and Group id

When you set up the Dspam CGI using the Apache web server, the CGI will always execute under a single user id. In order to deliver false positives to the correct user account, the LDA needs to change to the correct user id. Either Dspam or the LDA must be SetUID root to allow this context switch.

Dspam, Procmail, and Maildrop all have schemes to limit the exposure of running SetUID root, and SetGID in the case of Dspam. Because the CGI needs to be trusted to deliver mail to other user ids on the computer, you should do what you can to lock down Apache.

If the only reason you're running Apache is to provide access to the CGI quarantine, you can probably set it to a fairly standard configuration and have it be reasonably secure. If the CGI shares Apache with other web applications, you should install the mod_suexec Apache module and run it under its own user id. Before I get into actual Apache configuration, a quick note about user accounts.

Depending on your distribution and how Apache was installed, it may run by default as the nobody user, a user called apache, or perhaps something else like www or web. Exactly what user account Apache utilizes doesn't matter that much, but the effective user id can — on many systems, accounts with a user id greater than 65000 will not switch user context, even when executing a SetUID or SetGID program. The nobody account is often set to a number above this, limiting the damage that can be done if Apache or another process running as nobody is compromised.

Set up a real Unix user account with limited permissions for the Dspam CGI. On my system, I created a user account called dspamcgi. If the CGI is the only use of Apache on your system, you can simply set Apache to your user account with these two directives in `httpd.conf`:

```
User dspamcgi
Group dspamcgi
```

Whatever user you list here needs to be added to these places:

- `dspam.conf` Trust directive
- Unix group that Dspam is SetGID to, in `/etc/group` (the mail group in the default install)
- Maildrop's `trusted-users` configure switch.

Editing an htpasswd file every time you add a user to the system can quickly become a major hassle. There are several approaches to managing Apache authentication

CGI Setup

Besides creating a user account for the CGI script, you'll need to add directives to set Apache to execute the actual scripts. Copy the contents of the cgi directory from the Dspam source directory to a permanent place in the file system. I use `/var/www/dspam`. Then set up the appropriate alias and directory directives. For example:

```
Alias /dspam/base.css /var/www/dspam/base.css
Alias /dspam/logo.gif /var/www/dspam/logo.gif
Alias /dspam/dspam-logo-small.gif
   /var/www/dspam/dspam-logo-small.gif
ScriptAlias /dspam/ /var/www/dspam/
<Directory "/var/www/dspam">
Options +ExecCGI
  AuthName "DSPAM Quarantine Area";
  AuthType Basic
  Require valid-user
  Order Deny,allow
  Allow from all
</Directory>
```

The above settings make Apache find the scripts, stylesheet, and logos, and execute them using CGI. The `Order` and `Allow` directives relax the default IP address access controls to allow people to connect from any location. The `Auth*` directives set up authentication — but I haven't chosen an authentication method yet. The simplest way to get started is use the mod_auth module, included with the base Apache configuration. Create a password file using the htpasswd command:

```
# htpasswd -c /var/www/conf/htpasswd john
# htpasswd /var/www/conf/htpasswd seconduser
```

The first command, with the `-c` switch, creates the password file. The next line adds a user to an existing file. Then add the `AuthUserFile` directive to the `<Directory>` block above, near the other `Auth*` files:

```
AuthUserFile /var/www/conf/htpasswd
```

Once Apache is configured, check the configuration of the actual CGI. Add your user account to the admins file in the CGI directory, and edit the `configure.pl` file for the domain name of the web server and anything else custom for your configuration. Start Apache, and you should be in business!

Automatic CGI authentication

Editing an htpasswd file every time you add a user to the system can quickly become a major hassle. There are several approaches to managing Apache authentication. I recommend using mod_auth_shadow for local users, and mod_auth_mysql for virtual users. Both are available at SourceForge:

- mod_auth_shadow (`http://sourceforge.net/projects/mod-auth-shadow/`)
- mod_auth_mysql (`http://modauthmysql.sourceforge.net/`)

Both of these are Apache authentication modules, which are not included in the base Apache release. They can be easily added to an existing Apache installation. If you've compiled Apache from source code, you'll have everything you need. If you've installed from your distribution's package file, you may need to install the apache-devel or apache2-devel packages. Both of these modules use a script called apxs or apxs2 in the development packages or full source package to compile and load correctly.

mod_auth_shadow provides a way to authenticate against the built-in Unix user and group files. Modify the Makefile in the source distribution to point to the correct apxs script, and do:

```
# make all
# make install
```

You may need to add the module in the Apache configuration. To use mod_auth_shadow, simply change the `AuthUserFile` directive to:

```
AuthShadow on
```

mod_auth_mysql takes a bit more configuration, specifying the database, user, password, and several other settings. Read the documentation for more help.

Using a dedicated Virtual Host with suexec

The Apache suexec module makes it possible to specify a different Unix user id and group for different virtual hosts. If you run several web applications on the same server, you can limit access to the SetUID programs involved in Dspam by configuring a special virtual host to run as the trusted user.

Other virtual hosts on the same computer may still be able to access the programs, but if they're not trusted users, each program has a way to limit the risk associated with running SetUID root.

Dspam comes with a handy tool to generate spam training aliases, called dspam_genaliases. It parses /etc/passwd, finds all valid users, and outputs a list of users prefixed with "spam-"

To make your Dspam CGI run under Apache's mod_suexec, you'll need to install mod_suexec, specify the user id and group id to run as, and follow all the rules that suexec enforces to get your script to run.

Before you roll out mod_suexec, let me warn you about something. I personally make the virtual host for the Dspam CGI run under SSL encryption. The problem is, I have several web applications all running under SSL, but only one IP address assigned to the box. Because you can only have one SSL-encrypted virtual host per IP address, setting up Dspam using suexec on this virtual host means all the other web applications on that site must be set up under the same user accounts, and following all the suexec rules. The main one that has affected me is that suexec does not allow symbolic links — if you visit a URL that is actually a symbolic link to a script, suexec will not allow the script to execute and you'll get an error page.

That said, here's what needs to change to get mod_suexec to work:

1. Install mod_suexec. If you've installed Apache from your distribution's package system, you can probably install the appropriate RPM. If you've installed from source, try this configure line for Apache (make sure you have an Apache user, or change to reflect the main user account to run Apache):

```
./configure --prefix=/usr/local/apache2 \
   --enable-mods-shared=most \
   --enable-suexec \
   --with-suexec-caller=apache \
   --with-suexec-docroot=/var/www \
```

1. Change the main user and group in `httpd.conf` to apache or appropriate.
2. Put the remaining configuration directives (`Alias`, `ScriptAlias`, <`Directory`> block) listed in the section above inside a <`VirtualHost`> block.
3. Add this to the <`VirtualHost`> block:

```
SuexecUserGroup dspamcgi dspamcgi
```

1. Change ownership of the cgi directory and all the scripts in it to match the SuexecUserGroup:

```
# chown dspamcgi:dspamcgi -R /var/www/dspam
```

1. Restart Apache, and you should be up and running!

Conclusion

I've covered a lot of ground in this article. Running a mail server is no easy feat, and getting Dspam and its quarantine working correctly without losing mail can challenge the most diligent administrator.

If you take the time to trace the user id and group id at each step of delivery, you can sort out the issues and get it all to work.

Good luck!

Copyright information

© 2005 by John Locke

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is available at http://www.gnu.org/copyleft/fdl.html

About the author

John Locke is the author of the book *Open Source Solutions for Small Business Problems.* He provides technology strategy and free software implementations for small and growing businesses in the Pacific Northwest through his business, Freelock LLC (`http://www.freelock.com/`).

Poking at iTunes

A developer's guide to the iTunes platform

Chris J. Karr

One comment: *No wireless. Less space than a nomad. Lame.* Rob "CmdrTaco" Malda introduced the iPod to the Slashdot crowd with a statement rivalled only by Bill Gates' quip "640 KB should be enough for anybody".

Since that post in 2001, Apple's iPod quickly became one of the most successful products in consumer electronics history. While its success largely derives from its "hip" factor and stylish design, the iPod's integration with the iTunes music application and the iTunes Music Store has made the device a favorite among music listeners.

The iPod is a single component of Apple's larger music platform.

The hub of the platform is the iTunes desktop digital music player. The iTunes application is only one of many in a long series of digital music library applications. iTunes began existence as a classic MacOS application called SoundJam. When Apple began developing and promoting their digital hub strategy, Apple purchased SoundJam and rechristened it iTunes. Prior to the introduction of the iPod, iTunes was simply another Mac MP3 program.

In 2001, Apple released the iPod. The device differed from other portable digital music players because of the tight integration with iTunes. Since other portable digital music players connected to applications other than those provided by the devices' vendors, no other player was as tightly coupled with a desktop application as the iPod. While this close integration created a disadvantage where the iPod depends fully upon iTunes, Apple used the opportunity to craft the

Fig. 1: Both the music store and music sharing functionality are visible in this instance of iTunes

iPod as an extension to desktop-bound audio collections. Users can create static and dynamic playlists in iTunes, and the iPod synchronizes itself with those lists upon connection to the host computer.

While the iPod and iTunes originated as Mac-only products, in 2003, Apple expanded its strategy and ported iTunes to Windows and added USB functionality to the iPod. This opened the product's market to Windows users. In short order, the iPod and iTunes dominated the market for portable hardware players and digital music applications.

Later in 2003, Apple expanded the iTunes platform by introducing the iTunes Music Store. Previous online music

stores had struggled with balancing copyright protection and user convenience. But Apple made the iTunes Music Store a success by striking new deals with copyright holders and leveraging the tight integration with iTunes. Furthermore, the iTunes Music Store is only accessible from within iTunes and iTunes (via Quicktime) is the only desktop application legally allowed to play protected tracks purchased from Apple's online store. While critics had initially predicted the failure of the online music store, due to the difficulties inherent with Digital Rights Management (DRM) technology, the store succeeded and flourished due to the fine balance that Apple was able to maintain, between copyright holders and users. Apple is capable of maintaining this balance because it controls all levels of the technology platform used to acquire and play the protected tracks.

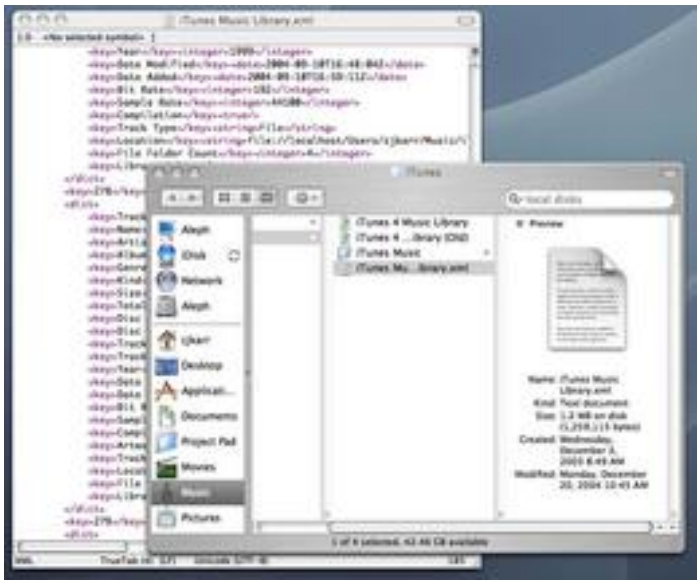
The iPod is a single component of Apple's larger music platform

In 2004, Apple further expanded the iTunes music platform with the introduction of AirPort Express base stations. The Airport Express device is a smaller version of Apple's Airport wireless access hardware. The Airport Express had also improved upon the older designs by including a dedicated USB hub, for printers and audio output ports, allowing connection between standard stereo equipment and wireless networks. To communicate with the Airport Express audio features, iTunes was given new functionality that sent audio to local Airport Express units. With this feature, iTunes users can use connected stereo devices to listen to their protected audio tracks. In addition to the analog stereo output port, Apple also integrated a digital 5.1 surround sound output into the device. (However there are currently no applications or content that can use the 5.1 surround sound feature as Apple has restricted access to third-party developers'.)

While Apple continued to expand the iTunes platform every year since its release, numerous third-party developers also created applications that integrate and interoperate with iTunes. Creative examples include music quiz applications that use the users' local iTunes libraries as a source for questions, and applications that assist users creating and refining metadata by providing connectivity to new data sources.

While Apple continues to restrict developer access to the Airport Express peripherals, it documented iTunes XML and AppleScript dictionaries for third-party developers to use when extending the iTunes platform. Furthermore, due to the efforts of persistent developers, undocumented portions of the platform (notably, music sharing protocols) are now better understood. Developers use this knowledge, official and otherwise, to create new applications that take advantage of the iTunes platform.

Fig. 2: The location and contents of the iTunes XML file



Programming with iTunes – XML, AppleScript, and Windows

Out of all the components in the Apple music platform, the iTunes application offers the most options for interested developers.

An XML file provides access to most of the iTunes metadata. Other applications can control iTunes with AppleScript (on the Mac) or the iTunes COM SDK (on Windows). Client and server applications can communicate with iTunes using the Digital Audio Access Protocol (DAAP). Finally, protected AAC files purchased from the online music store can be played using Apple's Quicktime API.

Writing applications that can read the iTunes library XML file provides the easiest route for adding new functionality. This file is located at the root of a user's music collection and is named "iTunes Music Library.xml". This file uses an Apple subset of XML. It begins with metadata about the iTunes application and continues with a list of track entries

and playlist definitions. The individual track entries contain metadata like creators, albums titles, and genres. The file also contains automatically generated metadata such as the last play date and local file locations. With the exception of album cover art, metadata encoded in individual tracks is also part of the XML file and is accessible to applications that read that file. Following the song entries, the file contains XML descriptions of the local playlists.

In addition to making its metadata available via XML, iTunes have also provided a rich AppleScript dictionary. MacOS X applications use AppleScript to automate common tasks or to control other applications. Users write AppleScripts in an English-like syntax that is designed for non-programmers.

AppleScript also provides access to frameworks such as Cocoa, allowing users to write standard GUI applications entirely in AppleScript. Furthermore, AppleScripts can be invoked from the command line or developed interactively with a bundled AppleScript interpreter called Script Editor. This application is a specialized environment for developing and testing AppleScript code. Developers access locally installed applications' AppleScript dictionaries from Script Editor. Using the iTunes dictionary, developers can discover the functions that the iTunes application provides via AppleScript.

Apple's approach to the DAAP developer community probably reflects a general uncertainty within the company about releasing the technical details of their popular applications

A script that instructs iTunes to start playing can be written in a single line:

```
tell application "iTunes" to play
```

With more code, finer control of the application is available:

```
tell application "iTunes"
       play track 13 of user playlist
         "Sparkle and Fade"
end tell
```

The AppleScript language also includes control statements:

```
tell application "iTunes"
       if player state is stopped then play
end tell
```

In addition to playback, iTunes also provides access to its encoding functionality:

```
tell application "iTunes"
       set preferred_encoder to current encoder
       set available_encoders to (name of encoders)
       repeat with anEnc in available_encoders
             if format of encoder anEnc is "AAC" then
                   set current encoder to encoder anEnc
                   exit repeat
             end if
       end repeat

       -- (Comment) do stuff

       set current encoder to preferred_encoder
end tell
```

Numerous resources are available online that describe in more detail the mechanics of iTunes and AppleScript.

In addition to Script Editor, AppleScript developers can run and compile their AppleScripts from the command line using the `osascript` and `osacompile` utilities.

These allow developers to use AppleScript in the same manner as traditional Unix scripting languages.

Finally, developers can use AppleScript within their own applications. Apple provides convenient access to AppleScript via the Cocoa API. In Objective-C, code for executing an AppleScript follows the form:

```
// asCode is a NSString object containing AppleScript
// instructions.

appleScript = [[NSAppleScript alloc]
                initWithSource:asCode];
[appleScript executeAndReturnError:nil];
```

Using these API calls, applications use AppleScript without the use of external script files.

When using AppleScript, there are many ways third-party applications can work with iTunes. A CD cataloging application can retrieve album data from iTunes and use that data to initialize a collection. Applications like iPhoto and iMovie use iTunes as a media resource for audio clips when creating home movies and DVDs.

Helpers to chat clients that query iTunes for the current track are common. Computer-based telephony applications can

Fig. 3: The iTunes AppleScript dictionary as seen from the Script Editor application

pause iTunes when an incoming call is answered and resume playing at the completion of a call. A particularly innovative application might use Apple's iSight web camera to determine whether a user is present and play some favorite music when a user is present.

Unfortunately, while iTunes on the Mac is accessible from an AppleScript standpoint, its Windows counterpart does not contain the same level of integration. This is due to the lack of a system-wide scripting architecture on the Windows platform. However, Apple provides a Windows-based Software Development Kit (SDK) for iTunes.

Unfortunately, while iTunes on the Mac is accessible from an AppleScript standpoint, its Windows counterpart does not contain the same level of integration

This allows Windows applications to interoperate with iTunes. The examples provided by the Windows SDK are Visual Studio Jscript files. The syntax of Jscript differs significantly from AppleScript:

```
// This code deletes tracks with empty locations.

var iTunesApp =
  WScript.CreateObject("iTunes.Application");
var mainLibrary = iTunesApp.LibraryPlaylist;
var tracks = mainLibrary.Tracks;
var numTracks = tracks.Count;

while (numTracks != 0)
{
       var currTrack = tracks.Item(numTracks);


       if (currTrack.Location == "")
       {
              // yes, delete it
              currTrack.Delete();
             deletedTracks++;
       }

       numTracks--;

}
```

While AppleScript remains the more popular tool, this may change in time given the growing number of Windows iTunes users and developers writing new applications using the iTunes COM SDK

Sharing and streaming with DAAP and iTunes

With the release of iTunes 4.0, Apple included music sharing between running iTunes instances on local networks. The original implementation allowed users to find shared libraries on local subnets and access remote libraries by entering the IP addresses of remote servers. In a later release, Apple removed the remote sharing functionality because of copyright concerns, but left the local sharing intact.

When the application is configured to enable sharing, iTunes announces the availability of the music service via Rendezvous. Other local clients receive this announcement and update their interfaces to display the newly shared library. When clients connect to the sharing hosts, the communication between the client and server uses the Digital Audio Access Protocol (DAAP).

DAAP is a protocol similar to HTTP that has been reverse engineered by Mac enthusiasts. While some of the programmers involved in documenting the protocol have received communications from Apple employees promising official DAAP specifications, presently no official documents exist. Despite the lack of documented specifications, the protocol is sufficiently understood that developers have successfully created servers and clients that interoperate with iTunes.

Developers desiring DAAP functionality within their applications can use a number of free software options. The `libdaap` library is used to create both clients and servers. The `daapd` server is used for sharing content. OpenDAAP

is another library for writing clients and servers, while the mt-daapd project produces a DAAP server that is multi-threaded and is less dependent upon external libraries.

For the developers wishing to work directly with the DAAP protocol, knowledge of HTTP is essential.

Not only does DAAP use a request format similar to HTTP, functions like user authentication use the same types of protocols and algorithms.

A sample DAAP request:

```
GET /server-info HTTP/1.1
Host: daap.example.com
Accept: */*
User-Agent: iTunes/4.0 (Macintosh; N; PPC)
Client-DAAP-Version: 1.0 Accept-Encoding: gzip
```

And the response:

```
HTTP/1.1 200 OK
Date: Thu, 01 May 2003 18:00:28 GMT
DAAP-Server: iTunes/4.0 (Mac OS X)
Content-Type: application/x-dmap-tagged
Content-Length: 122
Content-Encoding: gzip

[Byte stream continues\ldots{}]
```

While providing full DAAP documentation appears to be in Apple's best interest, the company has not yet committed to providing the full details of the protocol. This problem manifested itself during April 2004, when DAAP application authors discovered that Apple tweaked the authentication protocol from an MD5-based algorithm to a custom hash table algorithm. This broke numerous DAAP implementations and developers cried foul. Later, developers discovered that DAAP is a subset of a larger protocol called Digital Media Access Protocol (DMAP). DMAP provides portions of the photo-sharing capabilities in iPhoto 4 in addition to portions of the iTunes music sharing.

Apple's approach to the DAAP developer community probably reflects a general uncertainty within the company about releasing the technical details of their popular applications. Long criticized for its "not invented here" syndrome, Apple made progress establishing itself as a good-faith participant in the open standards and source community with technology like Darwin and Rendezvous. While reasonable people can be sympathetic to the company for withholding details in order to reserve a measure of freedom for later innovations, iTunes is a mature product and developers wish to write applications that extend the platform. If Apple decided to publish the details of DMAP and DAAP and then committed to making those standards open and public, companies such as Tivo who build complementary products to iTunes and iPhoto would be able to use existing protocols and standards rather than reinventing their own. (Tivo's sharing functionality is currently implemented using another program that utilizes another non-standard HTTP-based sharing protocol.) If third party product developers and consumers were confident that these standards were documented and stable, then software developers and equipment manufacturers could create extensions to the iTunes platform that would enhance its overall value.

In addition to Script Editor, AppleScript developers can run and compile their AppleScripts from the command line using the osascript and osacompile utilities

On platforms where market demand is not sufficient for motivating the creation of commercial products, free software developers would be free to participate and also contribute to the platform's aggregate functionality. Why Apple continues to refuse to document DAAP in light of these considerations remains a mystery. It may be an indicator that Apple plans to further change the DAAP protocol and continue to innovate, but it may also be a matter of simple institutional dysfunction lingering from its "not invented here" days.

Quicktime and FairPlay

While Apple's music store has been a success, it has not been without controversy. In order to reach distribution agreements with copyright holders, Apple is required to include digital rights management technology with every track sold.

The tracks sold on the iTunes Music Store are standard AAC audio files with embedded FairPlay protection technology. While Apple's DRM is considered less oppressive than competing alternatives, it still places restrictions upon the legal and reasonable use of the purchased tracks. For

example, while users can burn the tracks to compact disc, they are limited in the number of times a particular order of songs can be burned to multiple discs. Furthermore, while tracks can be shared among computers, users are limited to five computers that can be authorized to play the tracks.

AppleScript also provides access to frameworks such as Cocoa, allowing users to write standard GUI applications entirely in AppleScript

In response to these restrictions, a number of developers have created applications capable of removing the DRM components of the files.

This allows users to transcode the files in other formats and removes the limitations on playback. While this type of reverse engineering is illegal in countries such as the United States, these projects have continued underground by constantly changing the location of hosting servers and by distributing the software anonymously. For users in countries that prohibit the breaking or removal of DRM technology, applications can still be written that permit the legal playback of protected tracks. The drawback is that applications are required to use the Quicktime APIs and frameworks. Platforms without official Quicktime or iTunes implementations are excluded. This places a severe limitation on the users of free software operating systems, who wish to respect the copyright holders' rights while listening to their purchased music on alternative platforms.

Before Quicktime can play the protected tracks, the user must authorize the computer for playback using iTunes. After the computer has been authorized, Mac users can use the standard Cocoa and Carbon APIs to play protected files. To play a protected track using Cocoa, variations on the code below are used:

```
NSURL * url =
  [NSURL URLWithString:@"file:///path/to/file.m4p"];

NSSound * s =
  [[NSSound alloc] initWithContentsOfURL:url
byReference:NO];

[s play];
```

Similar functionality is available on Windows using the Quicktime SDK.

Fig. 4: The DAAPD home page (`http://www.deleet.de/projekte/daap/`)

While these methods provide playback functionality, they do not allow developers to access the raw audio sampling data and it's doubtful that this policy will change in the current copyright environment.

"Programming" the iPod

While developers can write applications that interoperate and use iTunes metadata, options for programming the iPod are much more limited.

While developers can write applications that interoperate and use iTunes metadata, options for programming the iPod are much more limited

While Apple does not offer a way to develop applications for the device, some functionality can be added to the device using its hypertext notes.

iPod notes use a subset of HTML tags and are limited to a size of four kilobytes. Titles can be set using a `<title>` tag, while line and paragraph breaks are created using the `<br>` and `<p>` tags. Links to files are created using standard `<a href="...">` tags. Note files that end in the `.link` prefix contain solitary links that automatically redirect to a particular note or song. Files ending in the `.linx`

extension appear to the iPod user as standard folders filled with links. This is not unlike directory pages generated by web servers when index files are missing.

In addition to the note files, the iPod also supports a museum mode that forces the device to boot into the note reader and prohibits the user from exiting the note reader application. This is often used to limit the user's access to other files and audio tracks on the iPod while presenting the audio in a proper context. This is useful in settings where the iPod is used as a supplemental educational tool.

For the developers wishing to work directly with the DAAP protocol, knowledge of HTTP is essential

While this is a simple format for text presentation that lacks many of the features normally associated with true application development, creative developers can produce the illusion of applications. This can be accomplished by using real desktop applications to generate a set of linked iPod notes that simulate a real application. For example, a developer can write a desktop application that reads weather data from an online source, generates a linked note set, and then copies those files to the iPod. To the iPod user, this would appear to be a genuine weather application. In another case, a read-only cookbook program can be constructed that generates a variety of subject and index notes that link to individual recipes. While this form of programming is much closer to static web development than real application programming, it does present creative developers with an opportunity to expand the iPod's basic functionality.

Conclusion

There are a number of contradictions for developers, within the iTunes platform. The application metadata and track information is easily accessible via XML. Audio sharing remains officially undocumented and developers find themselves at the mercy of future protocol changes. Robust AppleScript support is provided for fine-grained application control. However, Apple severely limits the ways and environments within which, protected audio tracks may be accessed. In some parts of the platform, Apple makes it simple to engineer compatibility and interoperability, while in other parts development is a black art fraught with peril.

On the basis of its advanced functionality and growing user base, the iTunes platform presents a tempting target for developers. Apple should be applauded for its efforts to encourage independent developers to embrace the platform, while at the same time it should be criticized for its lack of transparency and refusal to provide the necessary code and documentation that allow developers to create new players and network clients on platforms that are continually neglected by commercial desktop software vendors.

Bibliography

[1] Adams, Doug; Doug's AppleScripts for iTunes (http://www.malcolmadams.com/itunes/itinfo/info01.php), 2004

[2] Apple Computer; iTunes COM for Windows SDK (http://developer.apple.com/sdk/itunescomsdk.html); Apple Computer, 2004

[3] Apple Computer; iPod Note Reader User Guide (http://developer.apple.com/hardware/ipod/ipodnotereader.pdf); Apple Computer, 2004

[4] McArthur, Sandy; DAAP (http://sandy.mcarthur.org/DAAP); 2004

Copyright information

© 2005 by Chris J. Karr

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is available at http://www.gnu.org/copyleft/fdl.html

About the author

Chris Karr is the lead developer of the Books (http://books.aetherial.net/), a free software MacOS X application

Why free IT management tools are gaining traction

Enterprises are increasingly receptive to free software alternatives for IT management

Will Winkelstein

The $3.6 billion worldwide market for IT management software is ripe for competition from free software. Leading products from HP, CA, BMC and IBM are overkill for the vast majority of the market. Licensing costs can reach seven figures, and deployment and system administration costs are several times that. Not to mention that these products are widely known to be inflexible, monolithic and difficult to use.

At the same time, free software has penetrated close to three-quarters of all multi-billion dollar corporations and growth continues steadily. Industry research confirms that the primary reason IT organizations purchase free software solutions is the opportunity to reduce costs and improve technology performance. While leading products such as Linux, Apache and MySQL have generated the most attention, free software tools for IT management such as Nagios have matured and are poised for mainstream adoption. Since 2001, there have been 468,000 downloads of the Nagios program itself plus 221,000 for its plugins. Users include industry giants like ATT Wireless, Siemens, AOL, TicketMaster and TimeWarner Cable.

Combating inflexible platforms and vendor lock-in

Free IT management products are successfully competing against proprietary rivals — and for good reason. Proprietary IT management software tools and "platforms" such as HP OpenView and IBM's Tivoli are difficult to configure and resource-intensive to manage. Such IT management solutions have been designed for the top tier of enterprise customers. Feature-rich and unnecessarily complex, they are replete with functionality that just isn't needed by the vast majority of users. Increasingly inflexible and sclerotic, these IT management frameworks are hard to customize and difficult to configure. Deployment times can take months, even years, before value is achieved. And once they are installed, users face rigid vendor lock-in scenarios.

Fig. 1: GroundWork Monitor Alarm Report

Instead of being confined to a framework that precipitates add-on costs and maintenance fees (and limits future options), a free platform enables companies to easily link to-

gether existing monitoring tools and integrate new ones. Free IT management solutions can be customized to customer requirements given their configurable component architecture as well as their transparent, modifiable source code. They are also well suited to heterogeneous management tool environments, given their open interfaces. These free alternatives make an ideal "manager of managers" platform or can integrate as a peer to incumbent systems. They are also stable and reliable, having been broadly tested and peer reviewed.

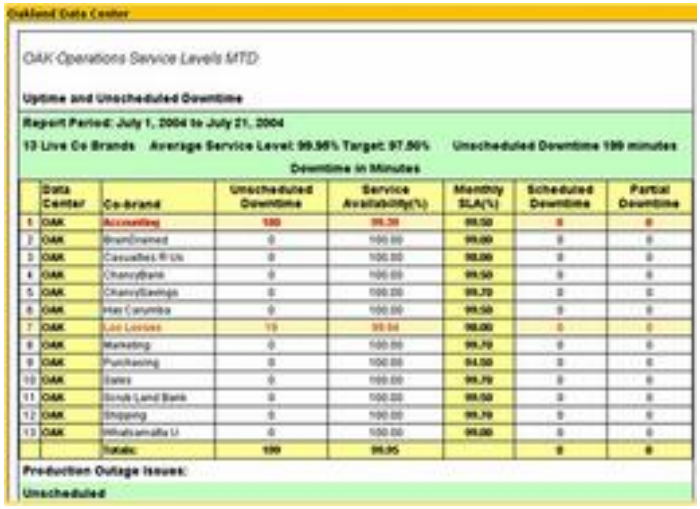
Lowering TCO

Enterprises have expressed strong interest in cheaper alternatives that meet their IT management needs. Today's proprietary IT management solutions typically require significant upfront license costs alone, with deployment and management costs running from $5 to $8 for every dollar spent on software. According to Forrester Research, 86% of companies indicate that low acquisition cost is the major reason for their decision to purchase free software, followed by 77% that express an expectation that they will lower their total cost of ownership. Likewise, according to Delphi Group, the biggest benefit driving the use of free software is total cost of ownership.

Free IT management solutions have proven to significantly reduce the total cost of ownership through efficiencies in several areas:

- **No licensing fees.** Because the software is free, customers pay only for enhancements, services and support.
- **Lower deployment costs.** With free software solutions, you don't install unnecessary features. This means that deployments are completed more quickly and easily. Companies save money by only paying for the features they need. Whereas a framework solution may run in the six figures and offer 100 different features, a company needing only 5% of that functionality can pay just for those requirements.
- **Low system administration overhead:** You only manage what you've installed. Free IT management solutions represent much lower administrative costs than proprietary alternatives. And because they are efficient without offering gratuitous features, they don't require expensive vendor-specific consultants and training.
- **Low hardware costs:** Free products typically run on inexpensive Linux boxes, further lowering overall costs.

Fig. 2: GroundWork Monitor SLA Report



A thriving free software ecosystem

While the IT management market finds itself faced with increasingly frustrated users of proprietary solutions, free software alternatives have emerged, which demonstrate highly competitive functionality. There are an abundance of free IT management products that stack up well against commercial alternatives, including Nagios for IT monitoring; RDTool for analytical graphing; Multi-Router Traffic Grapher (MRTG) for network device statistics; Nmap for network scanning and discovery; Ntop for network traffic analysis; SyslogNG for log file analysis; and Cacti for SNMP analysis and performance graphing. These products provide strong core functionality as well as some limited testing and support.

A growing ecosystem has also developed to help prepare the way for mainstream adoption of free IT management products. A number of companies offer the supporting elements for a complete solution: from usability, services, support, documentation and updates to integration, validation, testing and a sustainable knowledge base. SourceFire, for example, offers real-time network defense solutions based on free software. GlueCode provides a free Java applica-

tion platform. And my own company, GroundWork Open Source Solutions, offers a free monitoring tool as well as associated deployment, consulting and support services.

There are a significant number of free software projects dedicated to solving IT management problems. More than 70 consulting firms support Nagios. There are also a great many free integrators supporting free IT management. Europe has even more activity — no surprise given its track record as an early adopter of free software.

The tide is turning

More and more companies are opting to go with free IT management tools. There is also evidence of imminent defections among current users of commercial framework solutions. According to an as-yet-unpublished report by a respected IT research firm, 29% of CIOs and senior IT managers using HP OpenView, CA Unicenter, BMC Patrol or IBM Tivoli indicated they would consider switching to free software for IT management.

Free software incumbents in IT management demonstrate significant compatibility with technologies in the enterprise. They easily integrate with existing solutions without requiring dramatic changes to an IT infrastructure. And unlike many proprietary framework technologies, they are stealth by nature, enabling companies to incrementally embrace free software without necessitating major infrastructure modifications.

Copyright information

© 2005 by Will Winkelstein

Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Will Winkelstein is Vice President of Marketing at GroundWork Open Source Solutions (http://www.itgroundwork.com/), a leading provider of open source IT management solutions

ANSWER SQUAD.com

Tech Questions? We wrote the book!

Feel like roadkill on the Information Superhighway?

Questions about your computer, the Internet, the program you're writing, or even just the commercial application you're stuck working with every day? You've tried searching the Internet for solutions, with mixed results. You've asked people on various mailing lists, just to be flamed or answered with "rtfm!" You've tried tech support at Microsoft, Apple, Adobe or Macromedia, just to be overwhelmed by their cost. And here you are, still puzzled, still stuck, and still having to endure your computing environment rather than enjoy it.

Get roadside assistance with the AnswerSquad!

We're a group of best-selling authors and tech experts who are creating new, innovative and more efficient ways to help you find your computer nirvana: having everything work the way you want, when you want, and how you want.

Visit our website to learn how you can get the answers you need now.

AnswerSquad.com

Case study: Mythic Beasts

A small company specialised in Linux servers and amazing support

Tony Mobily

There is a company in the UK that provides Unix shells to their users: Mythic Beasts (http://www.mythic-beasts.com). They offer fantastic service to people who need a shell account on a very fast server, and don't want to fork out silly amounts of money. Let's talk to Chris Lightfoot, one of the company's owners.

TM: Who is behind "Mythic Beasts"? How did everything start?

There are six of us involved: Paul (see the photo), Richard, Pete, Tom, James and myself. Back in February 2000, I had the idea of setting up a colocated machine for our own use. I've dug out the original mail I sent inviting the others to get involved; it began,

- Plan: to host a computer somewhere, in order to have a decent platform to do web hosting, email, etc.
- Problem: cost.

At that time colocation was a bit more expensive than it is now! The first server itself cost us about £500.

Once we'd set things up for ourselves and our friends, we discovered that there was quite a lot of demand for Linux shell accounts, so (in August 2000) we registered as a UK limited company and Mythic Beasts Ltd. was born.

TM: Why the name "Mythic Beasts"?

The name "Mythic Beasts" is a complete accident. Our first server was called "mythic", because my convention is to name machines after adjectives and I thought the word "mythic" sounded good. That suggested the domain name "beasts"; when we came to set up the company - and remember that this was the tail-end of the dot.com era and lots of companies had *much* sillier names - we just recycled the name.

TM: Do you all work in the same office? Or do you normally work from home?

No, we work all over the place, mostly communicating by email and now IRC.

TM: Did you have a specific philosophy behind your idea? Did you decide your target market before hand?

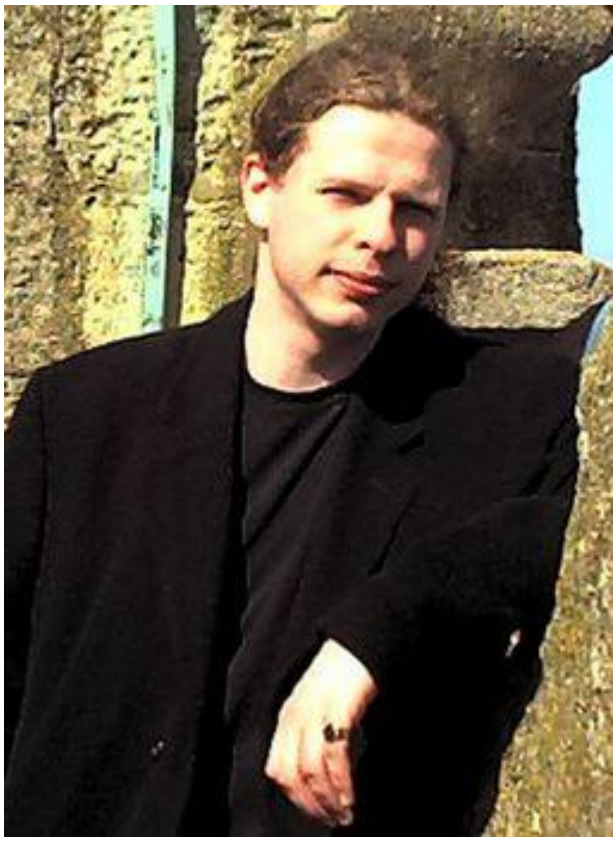
The way the company started was, as I've said, a bit of an accident - we spotted demand for something and were in a position to supply it. The reason we originally set up the services for our own use was that we'd all had bad experiences with shared hosting providers before, and wanted something better.

We don't have an official company philosophy or a mission statement which we engrave on paperweights (we don't even have a pointy-haired boss!) but I think the closest thing we can come up with is that we're all interested in selling services which we wouldn't be ashamed to use ourselves.

TM: What operating system (or systems?) do you run on those servers? How many people actively work on the server for maintenance?

At the moment everything runs Linux, mostly Debian but

Fig. 1: Chris Lightfoot



with some older (and heavily customised) RedHat installs. We're not zealots about operating systems (well, not much, anyway). But Linux does everything we need at the moment.

Most of the server maintenance is done by Paul, Pete and me, but unless we're installing new hardware, there isn't all that much to do. Modern PC hardware seems to be pretty reliable as long as (a) you pay enough for it to start with, and (b) you can hot-swap hard disks in your RAID arrays, since those are the components that fail most often.

TM: Who is your average customer? Do you ever have problems such as people getting an account to install an IRC bot, or to crack other servers?

Our typical shell account customer is interested in hosting one or several websites; we offer more flexibility than most hosting providers - for instance, our customers can write CGI scripts in more-or-less any language they want, can run web sites under FastCGI, etc. - and that's attractive for people who want a service which isn't going to get in their way too much.

We also have quite a few customers who use our services as an email service. Having a reliable shell account where you can run mutt or pine or whatever from anywhere in the world is a lot more convenient than running a desktop mail client and using web mail away from home (though obviously we support both of those options too). We also host some software developers who use our anonymous CVS services.

For users who want more sophistication we now offer a "virtual dedicated server" option, which is based on user-mode Linux; that's been quite popular for customers who want even more choice in what they can do, but don't want the hassle of owning and running their own hardware.

We've had occasional trouble with people buying accounts for IRC bots (which we prohibit, because this activity can be disruptive to our other customers), and also occasional individuals who try to buy accounts with stolen credit card numbers (apparently because this is a way for them to verify the numbers they've got). It's almost always easy to spot these kinds of account applications ahead of time.

TM: How do you organise your customer support?

Mail to support gets dropped into a home-brew ticketing system written by Paul (we used to use RT, but it was slow and overcomplicated). From there mail is distributed to a support list, which currently consists of five of the original investors. The ticketing system keeps track of who "owns" each support issue, and reminds us every morning about which tickets are still open.

TM: Is there a lot of custom software involved?

Yes, quite a bit, mostly written by Paul and me. You can see some of it on our software page at http://www.mythic-beasts.com/software.html (`http://www.mythic-beasts.com/software.html`) and more on our personal home pages. Some of the software we've written has found quite wide use: my POP3 server, tpop3d, is used by quite a few sites (some much larger than Mythic Beasts Ltd.), along with our virtual mail configuration tool, vmail-sql. Both of those were written back in 2000 because there weren't any suitable alternatives.

Paul wrote an excellent little tool called iftop for monitoring network performance in real-time, which I think is now in Debian and the *BSD ports.

Fig. 2: Paul Warren

TM: Do you release everything you wrote for Mythic Beasts as free software?

We don't always have the time to release things we've written - especially when they're very specific to our setup and therefore not likely to be of general interest.

TM: How important is free software for your company?

Companies like Mythic Beasts (not to say the internet as a whole) are only really viable because so much Free software is available, and we're glad to contribute what we can to the community.

TM: How much "politics" is involved in running a service like yours? For example, are you criticised for not having any Windows machines? Or for only having Linux? Have you lost any customers because of the SCO legal challenges?

We've encountered surprisingly little politics in running Mythic Beasts. As most of our users are familiar with Linux, and often have a strong preference for it, it is quite unusual for us to get requests for other operating systems. Almost all the services we run are based on open standards in any case, so we don't put any barriers in the way of people who use Windows on the desktop (which is probably the majority of our customer base).

We do have colocation customers who use a variety of operating systems.

As to SCO's recent legal challenges, we have not lost any customers due to this, and, if anything, have had more new customers since the challenges. We do not presently regard the whole SCO fiasco as a threat to our business (or to anyone else who uses Linux).

TM: What systems do you have in place for backup and disaster recovery?

On-site and off-site backups. All our customer-facing servers are hot-swap hardware RAID, backed by a support contract with our suppliers. The VDS servers back each other up, allowing us to restore a VDS on a different host within a few minutes - much better than most dedicated server arrangements.

TM: What are your plans for the future? I see that you are expanding your offer. Are you planning on growing considerably in the medium and long term?

Our plan is for steady growth with customer demand rather than significant expansion. We've invested a fair amount in new hardware this year and now have plenty of capacity for new customers.

We launched our VDS service which is proving very popular for a wide variety of applications that can't be handled easily in a shared hosting environment such as Java servlets and Subversion repositories, and for people who just want absolute control over the hosting of their website and email.

TM: Thank you very much for answering my questions!

Copyright information

© 2005 by Tony Mobily

Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Tony Mobily is the Editor in Chief of Free Software Magazine

Interview with Bernhard Reiter at aKademy

What we can do to promote the future of free software

Tom Chance

In his speech (http://conference2004.kde.org/cfp-userconf/bernhard.reiter-social.political.aspects.free.software.php) at aKademy (http://conference2004.kde.org/), Bernhard Reiter of the Free Software Foundation Europe (http://www.fsfe.org/) (FSFE) both celebrated Software Freedom Day and reminded the KDE community of what freedom in software means. The FSFE was founded in 2001 to promote and defend free software, and to coordinate national free software organizations, throughout Europe.

The FSFE was founded in 2001 to promote and defend free software, and to coordinate national free software organizations, throughout Europe

Bernhard is the Chancellor of the German Chapter of FSFE and has been involved since its inception. Interested in what he had to say to the KDE community, I caught up with him after his talk to ask him a few questions.

You mentioned in your presentation the threats of software patents and DRM (Damn Restriction Management, as you described it). What other areas do you think the free software movement will need to address in the next few years?

Well to begin with, the issues of software patents and DRM will be huge and so will be keeping us all busy for many years yet. Other issues do exist but they are generally ongoing as opposed to short-term projects. We expect that there will be more legislative and political attacks on free software, but we cannot predict what they will be.

The most important point I want to emphasize is that we must continue to educate people, and especially politicians, about free software We must convey what it means to the whole world, in terms of who controls technology and how it affects our lives. If we can do that, we won't have to spend so much time defending against attacks because more people will help us.

In Europe in particular, Governments are now driving a lot of the free software adoption. What part can projects like KDE and organizations like the FSFE play in this process?

Not everybody takes a huge interest in politics and I believe that it is normal that some people stay out of the technical side of things and just develop software. On the other hand we need more people to engage themselves politically for free software. Others should at least be responsible and keep their eyes open for events and developments where they can do their share. So they should, for example, be aware of demonstrations and lobbying efforts on issues like software patents.

In terms of development, members of the free software community should also bear in mind that code can shape laws.

Fig. 1: Bernhard Reiter

If the code doesn't allow a person to perform a particular action, then the code is controlling that person. In projects like KDE, which are reaching a fairly mature stage, developers should be focussing on the users and trying to make free software more usable for them. Even development can be political issue.

An important aspect of freedom in software is the ability to use it. But while KDE is currently translated into over 80 languages, there are hundreds more that are commonly used and that aren't yet covered. How can a community of volunteers, both in KDE and the FSFE, help this?

KDE developers already do help, in that they are part of a process of gradual improvements in internationalization and localization. In terms of free software organizations, apart from the FSFE in Europe and the FSF in North America there aren't any well established organizations.

We need more people to engage themselves politically for free software

We're hoping to change that, but we don't want to take money and help from governments and companies, even if it would help with internationalization, if it were to harm our independence and our ability to promote free software. We need to be careful about where we use our money, and what the specific focus of our investments are.

In general the direction that free software takes ought to be guided by competition and user demand, not interference from government

In general the direction that free software takes ought to be guided by competition and user demand, not interference from government

But don't you think that there is a role for government in areas where the market cannot yet supply full internationalization, especially where a large proportion of the population cannot speak English?

Well yes, of course government should be interested in preserving the cultural and linguistic heritage of their country. Where the market cannot provide, it is their duty to do so.

Moving back to software, are there any areas in particular on the free software desktop that you think developers need to address most urgently?

Fig. 2: KDE World Summit's logo

It is hard to say what is missing before we try deploying our software in the target markets, and so we will know a lot more when GNU/Linux is deployed in more corporations. I

am worried that many desktops are adding proprietary software to their systems where free software isn't yet available or good enough, usually because it is more convenient. Proprietary PDF readers and the MP3 format are an example where free alternatives exist (KGhostView and Ogg Vorbis, for example). Many proprietary tools are being used to develop free desktops; proprietary vector editing applications, for example, are used by many of the icon artists. We ought to be working to stop this being necessary or even desirable. Personally I would like to see more attention to business, with a good business model for private desktop users.

Personally I would like to see more attention to business, with a good business model for private desktop users

The FSFE has an initiative called the GNU Business Network (http://www.fsfeurope.org/projects/gbn/) which could help in this area if we received more funding. Society needs business, but business interests shouldn't drive it. Rather, society should be laying down the ethics by which business operates. Free software desktops can be part of the development of such an ethical business model for free software.

Should projects like KDE be more proactive in promoting free software then, including more protective licenses like the GNU GPL?

It's a matter of strategy. Yes, the KDE Project should promote freedom both by choosing free licenses (e.g. the GNU GPL) and by talking to partners and the public about it, but there are also times when it is more strategic to use a license with less protection (using the GNU LGPL, for example, to spread adoption).

The KDE Project ought to recognize how much it has given Trolltech, whose success is based to a large extent on the success of KDE

Qt on Windows remains a problem for us, since it is not available under a free license. KDE goes to some lengths to allow proprietary KDE software to be developed by releasing its libraries under the LGPL. This may be a good strategy - I don't want to condemn it in general - but it poses practical problems for KDE development, as they cannot accept GNU GPL code for the core libraries and the modules that require a Qt licensing exception when using GNU GPL. What code goes into which module within KDE is governed by this requirement that only benefits proprietary KDE applications.

The KDE Project ought to recognize how much it has given Trolltech, whose success is based to a large extent on the success of KDE. Maybe the KDE Project could encourage Trolltech to GPL the Windows version of its toolkit? The current arrangement is not unreasonable, but the KDE Project should be aware that it could talk to Trolltech on an equal level.

Bernhard, thank you for this conversation.

It's no problem; it is always good to educate more users about freedom. I would like to note that it is a good sign that the FSFE were invited to aKademy, and that the KDE Project and the FSFE have a good relationship. We welcome all future cooperation.

Copyright information

© 2005 by Tom Chance

This article is made available under the "Attribution-NonCommercial" Creative Commons License 2.0 available from http://creativecommons.org/licenses/by-nc/2.0/.

About the author

Tom Chance is a philosophy student, free software advocate and writer. He has worked in various guises with the KDE Project, the Association for Free Software, the Foundation for a Free Information Infrastructure and Creative Commons. You can contact him via his web site, http://www.tomchance.org.uk (http://www.tomchance.org.uk)

Hard passwords made easy

Creating strong memorable passwords using mnemonic devices and word lists

John Locke

In the online world, security plays a role in all online activities. Passwords are the most commonly used method to limit access to specific people. In my previous article (http://www.freesoftwaremagazine.com/free_issues/issue_01/passwd_management/) I discussed assessing the relative value of systems protected by passwords, and grouping passwords across locations with similar trustworthiness.

In a nutshell, don't bother creating and remembering strong passwords for low value systems, and certainly don't use the same passwords for low value systems that you use in high value systems.

In this article, I'll discuss how to create a strong password, and how to keep track of all your strong passwords, if you have a definite need to keep more than a couple.

In a nutshell, don't bother creating and remembering strong passwords for low value systems, and certainly don't use the same passwords for low value systems that you use in high value systems

Creating memorable strong passwords

A strong password is made up of several different types of characters, and isn't a name or word in a dictionary. Many systems that require strong passwords will check any password you try to create against a set of rules. These rules often specify a minimum length, and that your password includes characters from at least three of the following four groups:

Tab. 1: Character Groups

Capital Letters	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Lowercase Letters	abcdefghijklmnopqrstuvwxyz
Numbers	1234567890
Symbols	‘~!@#$%^&*()_+=-[]|}{‘;:”/.,<>?

The exact list of allowed symbols vary depending on the system. Some systems allow spaces in passwords, while others don't. A particular system might also have an international character set that includes other letters or characters.

Time after time, people forced to use strong passwords come up with some gobbledygook thing like “v7GT%Xz2.” Leave a computer to generate a password for you, and you could well end up with something like that. And the next thing that happens is they've forgotten it and need to call the administrator for a new one. It's certainly a strong password, but if you can't remember it, and don't store it in a safe place, it's not an effective password.

I suggest using one of three strategies for creating strong

passwords you can remember:

1. Create a password using a mnemonic device
2. Create a password using a word list with some variation
3. Create completely random passwords and store them securely.

Use a mnemonic device

Remember learning about mnemonics? Not Ebonics, that's something different. A mnemonic is a phrase or word to help you remember complicated or otherwise difficult to remember data. For example, ROY G BIV tells me the colors of the rainbow: Red, Orange, Yellow, Green, Blue, Indigo, and Violet - the letters in the name give you the sequence of the colors.

You can make up a phrase to remember a password, or make up a password based on a phrase that means something to you and nobody else

Jesus Christ Made Seattle Under Protest. No, not because there's so many heathen folk running about - this is a local mnemonic for remembering the order of downtown Seattle's streets, from south to north: Jefferson, James, Cherry, Columbia, Marion, Madison, Spring, Seneca, University, Union, Pike, Pine.

You can make up a phrase to remember a password, or make up a password based on a phrase that means something to you and nobody else. For example, our earlier "vhGT%Xz2" could become "Ve haven't Gotten Ten percent Hex sleep, too!" or some similarly silly meaningless phrase. Our brains are capable of easily substituting one symbol for another. I wouldn't trust this phrase for a password I only used occasionally, but for one you use several times a day, you'll remember it in no time.

For less-commonly used passwords, use a phrase with meaning to you, because you'll remember it easier: "Timmy and Tommy were my first dogs" could become "T&Twm1Dgs," which isn't a bad password at all. Remember your puppies and you've got your password.

Use a word list

A dictionary is a list of words. But I've already told you not to use dictionary words, right? Why is another word list okay?

Because you don't use just a single word, and you don't use a word that has personal meaning for you.

As a service provider, I often have to generate passwords for my customers. This is my favorite technique for doing that. You take a carefully generated list of words, and randomly pick two of them. Then you randomly pick a symbol or number to put between them. If it needs to be more secure, you then randomly make a few of the letters uppercase. Suddenly, you have a strong random password such as "rumpus!friar" or "fUngal)selMa." These can sometimes be quite amusing…

You can also add an element of fun to the actual password generation. Diceware.com (`http://diceware.com`) has two different word lists, and a method of randomly choosing words from them: by using regular dice. You scrounge through all those old board games in your closets to come up with 5 dice, roll them twice, and look up the word associated with the numbers you roll. Then you roll two of the dice to determine which number or symbol to put between them. Voila! You've got a reasonably strong password. I've found these passwords to be quite memorable.

Diceware is actually for creating longer passphrases, instead of passwords. A passphrase is used for encryption purposes, whereas a password simply provides access. Passphrases and encryption are a topic for other articles, but the passphrase generation ideas at Diceware make for a great way to generate passwords.

Store passwords securely

If you need to keep track of a bunch of different strong passwords, you have no choice but to record them somewhere. The problem is, where? Certainly not post-it notes attached to your monitor, or the bottom of your keyboard. I need to generate and store different strong passwords for many different clients. I don't want to remember them all, and I'm certainly not going to ask for them over e-mail, which has the security of a postcard.

If you're in this situation, you need a password vault of some kind, an encyrpted system that lists all of your passwords

and keeps them safe and secure. You still need to remember one password: the one that opens the vault.

I use a program on my Palm Pilot that stores all my passwords in an encrypted file. I can see all the accounts I've set up in the main screen, but to get the password, I have to enter the master passphrase first. After 5 minutes, the program automatically "forgets" the passphrase and re-encrypts everything.

I need to generate and store different strong passwords for many different clients. I don't want to remember them all, and I'm certainly not going to ask for them over e-mail, which has the security of a postcard

There are similar programs available for Windows and Pocket PC. You can also use generic encryption technologies like Gnu Privacy Guard (GPG), part of the excellent ()Wt software and provided in every Linux distribution.

Don't store your passwords in a plain text file, a Word document, an Outlook note, or a note in your PDA.

The important point is to think realistically about your risks. If your passwords are in a plain text file on your computer and it gets hijacked by a worm, virus, or attacker, your password file might get compromised without you ever realizing it. PDAs are incredibly easy to steal - you wouldn't want a thief to have instant access to all your passwords.

Password vault software

For Palm: Keyring for Palm OS (`http://gnukeyring.sourceforge.net`) is a great little free program that encrypts the password database to a password. The encryption is weak, but sufficient to protect your password for a few hours - if you lose your Palm, get a new one, restore your database, and change your passwords. The stronger your password, the longer a brute force attack will take. Also check out Strip (`http://www.zetetic.net/solutions/strip/index.html`) for better encryption, though its database is not viewable on your PC.

For Linux: A plug-in for Jpilot (`http://jpilot.org/`) can natively read the database for Keyring for Palm OS. This makes a great harmony: you can view, synchronize, and update passwords on both Linux and the Palm. Again, note that the encryption is weak, meaning the database can be cracked in a matter of 5 hours or so with brute force. That means you should protect your Palm backups, as well as Jpilot.

With these utilities and a better understanding of how to generate strong passwords you can keep your information safe from prying eyes

For Windows: Try Oubliette (`http://oubliette.sourceforge.net/`) or KeePass (`http://keepass.sourceforge.net/`), both are free software password managers for Windows. And here's another: Password Safe (`http://passwordsafe.sourceforge.net/`), developed by a well-known security expert, primarily written for Windows but with compatible versions for PocketPC and Linux available.

For PocketPC: There's a KeePass (`http://doncho.net/kppc/index.php`) version for PocketPC, too.

For Mac users, try Password Gorilla (`http://www.fpx.de/fp/Software/Gorilla`).

With these utilities and a better understanding of how to generate strong passwords you can keep your information safe from prying eyes.

Copyright information

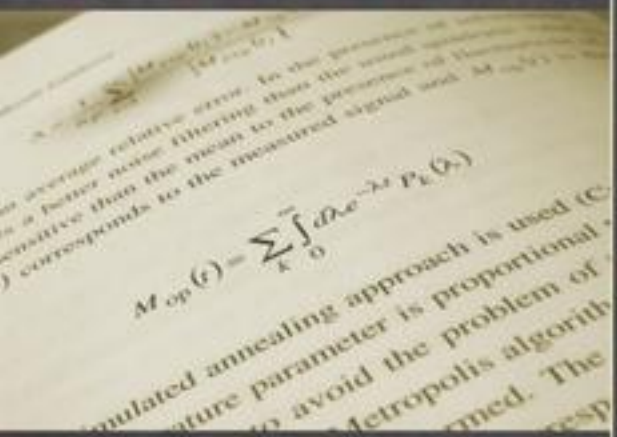
© 2005 by John Locke

Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

John Locke is the author of the book *Open Source Solutions for Small Business Problems.* He provides technology strategy and free software implementations for small and growing businesses in the Pacific Northwest through his business, Freelock LLC (`http://www.freelock.com/`).

Communicating mathematics...



...is getting easier.

Math Focus

News
Discussion
Collaboration
Software
Books

www.mathforge.net

Powerful tools for online mathematics

mathforge

qreate

Graphic, Web, Life design

PRINT

Free Software
MAGAZINE
CREATING FREE SOFTWARE MAGAZINE
A COMPLETE AUTOMATED SYSTEM TO CREATE MAGAZINES BASED ON FREE SOFTWARE
INTERVIEW WITH MYTHIC BEASTS' STAFF
THE SECOND MAC REVOLUTION
THE CONTENT TAIL WAGS THE IT DOG

FREE SOFTWARE MAGAZINE'S ISSUE 0 COVER

FREE SOFTWARE MAGAZINE'S MEDIA KIT

FREE SOFTWARE MAGAZINE'S ADVERTISEMENT

WEB

COLLE MORO BED&BREAKFAST
www.collemoro.com

VILLA DELLE RONDINI
www.villadellerondini.it

www.qreate.net ✻ *info@qreate.net*

REMIX READING

www.remixreading.org

Remix Reading is an artistic project based in Reading. It's aim is to get artists (working with music, images, text or video) to come together and share their work, be inspired by others' creativity, and to create"remixes". All material on the web site is released under a Creative Commons license which allows you to customise your copyright so others can use, copy and share your art as you choose. Please visit the website to learn more, download content or even add your own.

Do you want to download art for free without the threat of a court case? How about being able to copy and change it as you like?Are you an artist who wants to display your work in a creative forum?Do you want to share your work and allow people to remix it, whilst receiving credit?

email us at rreading@ibibilo.org

Read the full license for this poster on the Creative Commons website:
http://creativecommons.org/licenses/by-sa/2.0

Launch Event 1st March 5pm - 12pm

more info: http://wiki.remixreading.org/index.php/launchevent

James Sheridan
Campbell Orme

The FUD-based Encyclopedia

Dismantling fear, uncertainty, and doubt, aimed at Wikipedia and other free knowledge resources

Aaron Krowne

In this article, I respond to Robert McHenry's anti-Wikipedia piece entitled "The Faith-Based Encyclopedia." I argue that McHenry's points are contradictory and incoherent and that his rhetoric is selective, dishonest and misleading. I also consider McHenry's points in the context of all Commons-Based Peer Production (CBPP), showing how they are part of a Fear, Uncertainty and Doubt (FUD) campaign against CBPP. Further, I introduce some principles, which will help to explain why and how CBPP projects can succeed, and I discuss alternative ways they may be organized, which will address certain concerns.

Introduction

Recently, a friend of mine passed a rather noteworthy online article my way. The article, published in Tech Central Station, was entitled "The Faith-Based Encyclopedia," and was written by Robert McHenry [McHenry, 2004]. McHenry, the Former Editor in Chief of the Encyclopedia Britannica, was quite critical of Wikipedia in this article. Perhaps this comes as no surprise to readers who are already detecting the potential for a slight conflict of interest here. Still, I expected to learn something from this article, as Wikipedia is not perfect, and McHenry seemed like a reputable individual. Instead, I was greeted with an onslaught of FUD that left me flabbergasted. I can honestly say I learned nothing from "The Faith-Based Encyclopedia."

For the uninitiated, FUD stands for "Fear, Uncertainty, and Doubt." It is a term popular within the free software community, used to describe the use of lies and deceptive rhetoric, aimed chiefly at free software projects. It is an accurate term. In brief, the goal of FUD is to make money when the free software competition cannot be defeated fairly in the marketplace. This can be done by scaring consumers through wild propaganda, or more recently, confusing courts through more subtle arguments.

Foremost in the FUD hall-of-shame are figures such as Microsoft CEO Steve Ballmer and SCO Corporation CEO Darl McBride. CEOs such as Larry Ellison and Scott McNealy advance FUD from time to time as it suits them, but are not as single-mindedly unrelenting as the worst of the bunch. For more information on these gentlemen, their claims, and the truth, I refer you to web logs such as Slashdot, Lawmeme, and Groklaw.

McHenry's article is not so much remarkable for its own points (which I dissect below) as for the reaction to it. Most people, even those who know Wikipedia well, seem to lack confidence regarding its role in society. Even Larry Sanger, a former project member and co-founder of Wikipedia, largely defers to McHenry's sentiment [Sanger, 2004]. Sanger goes as far as suggesting Wikipedia needs to be more *elitist* (note how he doesn't say more *meritorious*), which would explicitly undermine the value of the project. I know other, very intelligent people, who had a "gosh, what were we thinking" reaction to McHenry's article, as if an angry god had come down from upon high to punish them for falling astray.

Fig. 1: Wikipedia: hated by former editors of print encyclopedias everywhere?

Fig. 2: FUDslingers: (from top-left, clockwise) Microsoft's Steve Ballmer, SCO's Darl McBride, Sun's Scott McNealy, Oracle's Larry Ellison, and now (pictured center), Britannica's Robert McHenry?

Some of this attitude is understandable - Wikipedia is so new that most people don't know what to make of it. However, people like McHenry, with a vested interest or deep-seated bias, are ruthlessly taking advantage of this trepidation. I would implore everyone to weigh their own experiences in the matter much higher than the abstract arguments of third parties.

In the case of McHenry, I believe that these arguments are a new kind of FUD, aimed not just at free software, but at any form of free or open resource. I will make the case that there are in fact no terminally fatal shortcomings in Wikipedia, and that Robert McHenry is a pioneer of this new world of FUD, rightly deserving a spot next to Ballmer and McBride in the FUD pantheon.

Introduction to Wikipedia

Wikipedia is a collaborative, internet-based encyclopedia, made up of individual (but interconnected) *entries*. Anyone can create an entry for the encyclopedia, and anyone else can come along and edit existing entries. The intent is to tap the power of the general internet community's knowledge and the desire to share that knowledge, to build a free, high-quality, comprehensive online encyclopedia.

At first blush, it might seem like this could not work, and that the result would be a chaotic mess. But it does work. This is true due to a number of features of the collaborative system (mediawiki) and underlying social dynamics. I will discuss these later.

Wikipedia is a good resource. In fact, it is an incredible resource. As a frequent user of Wikipedia, I believe I am qualified to make this determination. Hardly a day goes by without my accessing a handful of Wikipedia articles, either from a web search, forwarded by a friend spontaneously or in reference to a conversation we have had, linked from other Wikipedia articles, or accessed directly in an ad hoc fashion.

I have noticed recently that Wikipedia is being invoked more, and contains more articles of a higher quality than in the past. Further, I cannot remember the last time I accessed a Wikipedia article that was not of apparent professional quality. Indeed, Wikipedia is almost becoming authoritative, a fact which clearly upsets McHenry and similarly-situated individuals.

I have never used an encyclopedia as much as Wikipedia and I thank the Wikipedia community for what they have created. Countless others share these sentiments. Wikipedia has enhanced my life and brought considerable *progress* to society. I consider these facts so easy to demonstrate that they are pointless to debate.

It is into this milieu that McHenry's article arrives. Suf-

fice it to say, I find his attitude and claims a little surprising, given the above observations. Those I know who are involved with Wikipedia do not seem to have received the same memo on the imminent collapse of their project. But as Görring once said, the bigger the lie, the more people who will believe it. I think this is what McHenry is shooting for.

Response to McHenry

McHenry's central thesis is that, quite contrary to general observation, Wikipedia is a poor-quality resource, that it is in a constant state of chaos, and that these problems will tend to get worse over time. Of course, he doesn't explain how one is to reconcile this claim with the increasing popularity of Wikipedia, other than a veiled suggestion that people are simply stupid.

McHenry begins his article with a proof-by-denigration. He takes the entire proposition, that a commons-based collaborative encyclopedia could even be successful, as ridiculously out-of-hand. He recounts Wikipedia's failed first start as "Interpedia," which I presume is done to poke fun at the concept, the people, and the community process behind Wikipedia. Then, in heavily loaded terms, he characterizes the claim of increasing quality as (emphasis added):

> Some *unspecified quasi-Darwinian process* will assure that those writings and editings by contributors of greatest expertise will survive; articles will eventually reach a steady state that corresponds to the highest degree of accuracy.

He then goes on to say:

> Does someone actually believe this? Evidently so. Why? It's very hard to say.

Actually, I don't believe it is so hard to say, and will go into detail on the matter shortly.

McHenry then goes on to theorize that the currently-vogue educational technique of "journaling" is responsible for corrupting the thinking of today's youth, consequently leading them to believe in something as ridiculous as the success of collaborative commons-based projects. I can only surmise that this bizarre tangent is due to a pet peeve of his, and believe it can be safely discarded.

Our friendly author then takes a stab at empirics. His method is to sample a number of Wikipedia entries, inspecting their previous versions and revision history to ascertain whether the quality has increased or decreased. The size of McHenry's sample set is: 1.

At this point, it is worth noting that Wikipedia recently reached the 1 *million* article mark.

About this dubious empirical method, McHenry says:

> ... I chose a single article, the biography of Alexander Hamilton. I chose that topic because I happen to know that there is a problem with his birth date, and how a reference work deals with that problem tells me something about its standards.

So in other words, he is quite cognizant of the fact that he is about to make an induction from one article to *one million*, based on *a degenerate case*. I will temporarily leave the reader to make their own value judgment of this policy, and proceed within the bounds of McHenry's game.

McHenry finds two problems with the Hamilton article. The first is that it gets a date wrong. The second is that it has declined in quality over time, at least, according to his standards. McHenry's definition of quality seems to consist solely of presentational matters such as spelling, grammar, and text flow. These are of course important considerations, but I propose that there are other important facets of quality - for example, coverage. In a later section, I will attempt to sort out some of the confusion on this topic. For now, let us again take McHenry's claims at face value, and proceed to the finale.

He concludes with a metaphor. I will reproduce it here in full (emphasis mine again):

> The user who visits Wikipedia to learn about some subject, to confirm some matter of fact, is rather in the position of a visitor to a public restroom. It may be obviously dirty, so that he knows to exercise great care, or it may seem fairly clean, so that he may be lulled into a false sense of security. *What he certainly does not know is who has used the facilities before him*.

Fig. 3: "Historical reviewability akin to a public restroom" describes traditional encyclopedias, not CBPP ones like Wikipedia.

Fig. 4: Viewing past corrections and changes are not exactly facilitated by this style of encyclopedia (but Wikipedia contributors have, admirably, not slandered them in spite of this).

McHenry is essentially asking us to suspend all higher brain function at this point: in the above metaphor he simply pretends that the reviewability features *he just based his entire analysis on* do not exist.

A reminder is perhaps in order: to determine that his not-so-randomly chosen article declined in quality, McHenry used Wikipedia's revision history feature to look at how it had changed over time. This feature of Wikipedia, which is a hallmark of open content production systems, makes it precisely *the opposite* of a public restroom. You *can* in fact see everything that "came before you" with Wikipedia.

What would McHenry's metaphor apply more fittingly to?

Why, a traditional print encyclopedia, of course. If I wanted to analyze an arbitrary Britannica article's evolution over time (for example), I'd have to somehow acquire the entire back catalog of the Britannica (assuming older editions can even be purchased), presumably reserve a sizeable warehouse to store them all, and block out a few days or so of my time to manually make the comparison.

Even the electronic forms of traditional encyclopedias are sure to be lacking such reviewability features. This makes sense, as public reviewability would be *embarrassing* to traditional content creators.

So, in an artistic twist of doublespeak, McHenry has attempted to convince the reader that one of the key failings of traditional, closed media is actually the main problem with open, collaborative content. Presumably, love is also hate, war is peace, and so forth.

As if this was not enough, there were some other asides and general themes of his article that scream "FUD," or simply boggle the mind by virtue of their illogic.

For example, McHenry makes a point that seems like it should, by all rights, completely discredit his own article. He says (emphasis added):

> I know as well as anyone and better than most what is involved in assessing an encyclopedia. I know, to begin with, that *it can't be done in any thoroughgoing way. The job is just too big*. Professional reviewers content themselves with some statistics – so many articles, so many of those newly added, so many index entries, so many pictures, and so forth – and a quick look at a short list of representative topics. Journalists are less stringent.

So in other words, *no one* can conclusively assess an encyclopedia. This is an odd thing to say before proceeding to assess an encyclopedia. McHenry also fails to meet even the approximate standard of his "professional reviewers," unless you seriously consider examining one article to be statistically significant. Personally, I think routine use, amortized over millions of people, can actually shed some light on quality.

An important underlying theme of McHenry's piece is his repeated harping on the fact that you "never really know" if a Wikipedia article is true, and his jeering at the Wikipedia community for honestly admitting this. Simultaneously, he utterly ignores the fact that *this is also the case for traditional encyclopedias*. Once again, this is a purely rhetorical device, as no one in their right mind would claim that traditional encyclopedias are perfect.

I am of the school of thought that when you critique something for which there are alternatives, you should apply the same review criterion to *the alternatives* as you do to the main subject. Perhaps McHenry is just not a student of this school.

To sum it all up, the majority of McHenry's piece is laced with snide, pejorative, and in general, loaded words and statements. Many of his points undermine his core claims, but his argument-through-sentiment goes a long way towards leaving a negative impression of Wikipedia on the reader. This is a shame, and I hope that few people are dissuaded from using or contributing to the resource for this reason.

In the next section, I will address the following *serious* (but only implicit) claims of McHenry:

1. Individual article quality in CBPP systems will inevitably (and monotonically) decline.
2. The quality of the entire CBPP resource will decline.
3. There is only one facet of quality that matters.

Understanding CBPP

Commons-Based Peer Production refers to any coordinated, (chiefly) internet-based effort whereby volunteers contribute project components, and there exists some process to combine them to produce a unified intellectual work. CBPP covers many different types of intellectual output, from software to libraries of quantitative data to human-readable documents (manuals, books, encyclopedias, reviews, blogs, periodicals, and more).

Examples of successful CBPP efforts abound. The Linux kernel, the GNU suite of software applications, and the combined GNU/Linux system are prime examples of software CBPP. Slashdot.org is an important example of CBPP through submission of the news articles, and more importantly, the collaboratively-based comment filtering system. Kuro5hin.org is an example of a collaborative article and current-events essay blog, with an emphasis on technology and culture. Wikipedia is an example of a comprehensive encyclopedia.

Even commercial sites, such as Amazon.com, have significant CBPP elements nowadays. These manifest in Amazon as user-submitted reviews and ratings and interlinked favourites lists.

The enabling dynamic of CBPP is that people are willing to volunteer a little bit of work and a large amount of knowledge to online community systems, and that when this force is properly harnessed, significant overall value can be created.

This trend has not gone unnoticed, or unexplained.

In fact, the term CBPP was coined by Yochai Benkler in his seminal work "Coase's Penguin: Linux and the Nature of the Firm" [Benkler, 2002]. In this work, Benkler acknowledges that CBPP is a real and common phenomenon underlying many important intellectual efforts in the world today, and attempts to answer "why" and "how." Benkler explains why CBPP happens in terms of a simple economical analysis, which reveals that CBPP is a new mode of production, existing alongside markets and firms as a productive modus operandi.

However, I think there are some details about how CBPP functions that have not been summarized anywhere. Accordingly, I'd like to propose a pair of laws, which I believe apply to collaborative, commons-based production, and explain why Wikipedia and other CBPP efforts succeed. We could call them the "first two laws of CBPP:"

- (Law 1.) When positive contributions exceed negative contributions by a sufficient factor in a CBPP project, the project will be successful.

This factor probably has to be greater than two in practice, because each negative contribution needs a proportional amount of work to be un-done. In other words, the project success criterion is $p > an$, where $a > 2$, and p and n represent positive and negative contribution quantity, respectively.

My guess for an approximate, "universal" minimum value of a would be about 10. This value probably has to be much greater than the theoretical minimum (2), because people in social settings will be annoyed and demotivated by a smaller quantity of adversity than is actually insurmountable. In general, contributors don't want to be spending a significant amount of their time treading water and backtracking. For example, if $p/n = 3$, then 33% of positive contributions are not going towards permanently improving the product.

However, I would bet that the positive-to-negative contribution ratio in many collaborative projects exceeds levels of 100 or 1000. In principle, the approximate value could be discovered empirically for particular projects. This would be a very interesting study to undertake.

- (Law 2.) Cohesion quality is the quality of the presentation of the concepts in a collaborative component

Fig. 5: A cartoon illustrating the relationship between positive and negative contributions and their effect on overall collaborative project success. The precise meaning of *a* in this picture can be thought of as the ratio of the weight of negative contribution "bricks" to their positive counterparts, times the ratio of the quantity of each type of contribution.

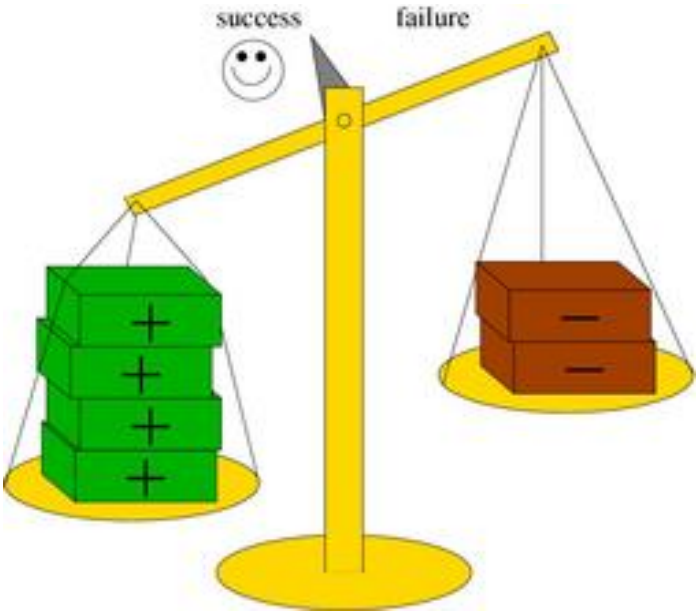


Fig. 6: A hypothetical chart of the change in quality of a collaborative object, with edits over time. Note that quality is normalized at 1, as we can model each object as having a theoretical "perfect" delivery and "complete" coverage (though it would be difficult to demarcate such objective levels in practice). Note that coverage quality almost never declines.

(such as an encyclopedia entry). Assuming the success criterion of Law 1 is met, cohesion quality of a component will overall rise. However, it may temporarily decline. The declines are by small amounts and the rises are by large amounts.

This phenomenon can be explained by the periodic, well-meaning addition of new conceptual materials by disparate individuals. These additions are often done without much attention to overall presentation (i.e., pedagogy, terminology, flow, spelling, and grammar). However, when these problems become large enough (with respect to a single project component), someone will intervene and make a "massaging edit" that re-establishes the component's overall cohesion.

To this I also add a companion corollary:

- (Corollary.) Laws 1 and 2 explain why cohesion quality of the entire collection (or project) increases over time: the uncoordinated temporary declines in cohesion quality cancel out with small rises in other components, and the less frequent jumps in cohesion quality accumulate to nudge the bulk average upwards. This is without even taking into account *coverage quality*, which counts any conceptual addition as positive, regardless of the elegance of its integration.

I think these laws take a lot of the mystery out of Wikipedia. The intent here is not to diminish the achievement, but to help us understand how it can be so. Thoughtful people tend to expect a tragedy of the commons when considering any commons-based production effort, but CBPP defies this expectation. I have learned to accept and embrace this reality, as have impartial observers such as Benkler. Others, such as McHenry, have not. Perhaps this conceptual framework would help them move on.

Other Styles of CBPP

Wikipedia does not embody the only model for CBPP. In fact, there are numerous models, corresponding to the type of intellectual output being produced, as well as the characteristics of the specific community producing it. I believe it

Fig. 7: PlanetMath is an example of a CBPP project, which is organized in a way that addresses concerns about integrating expertise.

is important to distinguish between CBPP and specific models of CBPP, so as to allow people to make an educated decision as to whether to employ CBPP for their efforts. What is good for Wikipedia may not be the best model for your project.

McHenry spends a lot of time in his article bemoaning the fact that anyone can edit a Wikipedia entry. While it is obvious that this modus operandi has not prevented Wikipedia from becoming a great success, it's not necessarily so obvious that there are other ways to organize CBPP. So even if this was a legitimate problem, it still would not herald the downfall of CBPP.

In previous work [Krowne, 2004], I identified one of the key attributes of CBPP as the *authority model*. The authority model of a CBPP system governs who has permissions to access and modify which artifacts, when, and in what workflow sequence.

In that study, I outlined two authority models, the *free-form model*, and the *owner-centric model*. The free-form model, which Wikipedia employs, allows anyone to edit any entry at any time. Changes can of course be rolled-back, but they can also be re-applied. The ultimate guard against malicious behaviour is therefore the administrators of the site. However, ne'er-do-wells typically lose interest long before it is necessary to resort to administrative powers.

My own project, PlanetMath, employs the owner-centric model by default. In this model, there is an *owner* of each entry - initially the entry's creator. Other users may suggest changes to each entry, but only the owner can apply these changes. If the owner comes to trust individual users enough, he or she can grant these specific users "edit" access to the entry.

These two models have different assumptions and effects. The free-form model connotes more of a sense that all users are on the "same level," and that expertise will be universally recognized and deferred to. As a result, the creator of an entry is spared the trouble of reviewing every change before it is integrated, as well as the need to perform the integration. By contrast, the owner-centric authority model assumes the owner is the de facto expert in the topic at hand, above all others, and all others *must* defer to them. Because of this arrangement, the owner must review all modification proposals, and take the time to integrate the good ones. However, no non-expert will ever be allowed to "damage" an entry, and therefore resorting to administrative powers is vanishingly rare.

It is likely that these models have secondary consequences. A natural result of the free-form model may be that entries lack some cohesion, and perhaps may even be of lower overall quality, despite having high coverage. On the flip side of the coin, the owner-centric model can be expected to foster a high level of cohesion quality, but may actually lag in coverage due to higher barriers to individual contribution.

McHenry's concerns can be translated into this framework, in which case he seems to be worried about the free-form model's effect on entry cohesion quality. (There's another interesting effect here: coverage quality is potentially less-bounded in digitized resources than in print ones, due to the near-complete removal of physical space limitations. Someone who only understands print resources - or has a vested interest in them - would therefore naturally de-emphasize the importance of coverage quality.) I don't believe this is a legitimate concern for Wikipedia, given the success of the project and utility of the overall resource. In fact, in the study I conducted in [Krowne, 2004], I discovered that while concern about the free-form model was common, it was not clear that it had any negative impact on productivity.

Instead, it seems likely that the multiple facets of quality (cohesion vs. coverage, and perhaps others) play into a balance in overall entry quality, relative to the different possible authority models. McHenry's concerns about cohesion quality might be more legitimate in other settings, or given

a different balance of expertise, time available to volunteer, or malicious intent present. I believe further work needs to be done to study the issue of quality and CBPP authority models, as I suggested in [Krowne, 2004].

Conclusion

McHenry's piece is full of so many contradictions of logic and sentiment that it is perhaps best understood (if you want to be charitable) as the train-of-thought reactions of someone personally upset by change.

> You "never really know" if a Wikipedia article is true... but *this is also the case for traditional encyclopedias...* no one in their right mind would claim that traditional encyclopedias are perfect

According to McHenry:

- Wikipedia has achieved much (this is difficult to deny). On the other hand, it barely got started (hah!), presumably because of its CBPP nature.
- There are no truth-guarantees with any knowledge resource. On the other hand, Wikipedia is all junk written by high-schoolers, even when its content is not patently false.
- Wikipedia is honest and forward with its caveats (again, difficult to deny). On the other hand, it needs caveats (hah!).
- Clearly the popularity of Wikipedia is astounding and people are abandoning print encyclopedias for it and other internet resources. On the other hand, people are mindless sheep who will accept junk, so long as it is free.

If in fact intentional and premeditated, this article amounts to propaganda-like doublespeak that would make Orwell blush. The intent seems to be to leave people with a negative feeling about Wikipedia and other CBPP resources. Such an intent is difficult to prove *prima facie*, since McHenry mentions enough positive fact about Wikipedia to give himself and his ideological allies a basic line of defence. However, this defence crumbles upon further inspection.

In this article I have addressed McHenry's actual points and implied complaints about CBPP, as well as introduced some descriptive principles of CBPP dynamics that help explain why the mode of production works. I hope these things help put the misconceptions and falsities propagated by McHenry and his ilk to rest. If this kind of FUD is required for traditional encyclopedias to survive in the future, then they will have rightfully earned the title "FUD-Based Encyclopedias."

Bibliography

[1] Benkler, Yochai "Coase's Penguin: Linux and the Nature of the Firm," Yale Law Journal, December 2002. online article (http://www.benkler.org/CoasesPenguin.html)

[2] Krowne, Aaron, Bazaz, Anil "Authority Models for Collaborative Authoring,"' HICSS 2004 Proceedings, January, 2004. online article (http://br.endernet.org/~akrowne/my_papers/authority_models/authority_models.pdf)

[3] McHenry, Robert "The Faith-Based Encyclopedia," Tech Central Station, November 15, 2004. online article (http://www.techcentralstation.com/111504A.html)

[4] Sanger, Larry "Why Wikipedia Must Jettison its Anti-Elitism," Kuro5hin.org, Dec 31 2004. online article (http://www.kuro5hin.org/story/2004/12/30/142458/25)

Copyright information

© 2005 by Aaron Krowne
Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Aaron Krowne runs PlanetMath.org and serves as Head of Digital Library Research at Emory University's Woodruff Library. He holds a M.S. in Computer Science from Virginia Tech.

Guerrilla marketing

Part one: promoting community projects in the marketplace

Tom Chance

It is a common assumption that companies who distribute free software will promote it, leaving the community to concentrate on the meat of the project itself (including code, documentation, graphics, and so on). But this is untrue; companies generally devote few resources and little expertise, leaving communities to fend for themselves in the big scary world of media and marketing.

Thankfully, all is not lost. Projects such as Mozilla Firefox, KDE and GNOME are fighting back with increasingly sophisticated promotion teams. In the past year we've seen massive fund raising initiatives, ever increasing community press coverage and a blossoming free software media market. If you're in an under-promoted project, or you're in a position to help, then this article is for you.

Community or marketplace?

Most free software projects operate in non-profit communities whose products are commercially distributed in the marketplace by third-party companies. This relationship has always benefited both parties, with most work done by one party benefiting the other. But this isn't the case with marketing, where in general neither the communities nor their commercial distributors promote the products directly.

From the distributors' point of view, this isn't useful. They're in the business of differentiating themselves from the GNU/Linux-based competition, and most free software projects will be used, without significant modifications, by all of their competitors. Instead they focus on promoting the value that they add to the projects' products, and to a lesser extent the major features that make the project stand out from Microsoft Windows and Apple MacOS. So community projects do get promoted, but only indirectly.

For communities, a lack of people, resources and expertise tend to mean that promotion work is neglected. Time spent developing a watertight marketing strategy tends to be spent instead fixing bugs and implementing features, which is a reasonable compromise. The problem, then, is whether or not communities can effectively promote themselves in the marketplace. Even if your project attracts a few people with the requisite skills and you have the finances to print materials, attend trade shows and pay for advertisements, there is a wider problem: is your project commercially comprehensible? Will a company or public body be attracted to your community because of your adherence to free software principles, and will they be interested in learning about how they can interface with your volunteers? These hurdles aren't insurmountable, and as the case studies I will examine will show, there is plenty that a community can do.

The spread of Firefox

Enter the fox. With relatively little company backing, a product from a major free software community achieved one of the most remarkable marketing successes of the year. The community headquarters, Spread Firefox, gained over 50,000 volunteers in a little under four months, and gener-

ated 13 million downloads in only 45 days. There won't be many people in the IT industry this year who won't have heard of Mozilla Firefox.

The community's technique was based on word of mouth, far more persuasive than any advertising campaign. From the early days of Phoenix and Firebird (Firefox's previous incarnations), enthusiastic users were already telling their friends and family to try the web browser out. These users naturally developed shared beliefs about the benefits of using Firefox, and values relating to free software and open standards, which provided the basis for a rare commodity in the free software world: a genuine community of users. But it was still just an undeveloped basis, when in reality they had been a loosely connected fan club.

The first change came with an overhaul of the Mozilla web site: The homepage for each product presented a simple, coherent frame that fans could adopt. The frames were ideologically congruent with the fans, meaning that the fans passionately agreed with the spirit of the project presented by the frames, and the frames presented a short list of features that were popular with the fans (focusing on tabs, pop-up blocking and security). As fans began to consolidate around this shared message, a nascent community of marketers developed.

In September 2004, five enthusiasts announced Spread Firefox (sfx for short), a hub around which volunteers could gather to discuss and promote their favourite product. Sfx introduced three new techniques: targets, incentives and tools. From their first target of one million downloads in ten days, they expanded to other download figures, as well as raising funds for a full page advert in the New York Times.

To spur volunteers on, they cleverly presented simple incentives like web referral points (for each click on an advert), a regularly updated volunteer spotlight and plenty of community tub-thumping. Finally, they provided volunteers with consistent branding by way of advertising buttons for web sites, email signatures, leaflets, posters and literature. Armed with these tools, volunteers were able to spread the word more quickly and consistently than in any previous free software community effort.

Sfx matured with time, creating a wide range of positions for the most keen volunteers involving programming, writing materials, acting as a press contact, designing adverts, handling CDs, attending events and posing as college reps. Some teams worked on specific target markets such as colleges and home desktop users, tailoring materials to suit their audience.

Fig. 1: Spread Firefox offers a range of targets, tools and incentives

They cleverly presented simple incentives and tools and were able to spread the word more quickly and consistently than in any previous free software community effort

The most spectacular success was the New York Times advert. The team set out to raise US$50,000 to finance a full page advert, and provided an incentive by offering every donor a space in the advert for their name. By the time the advert was in print they had raised over US$250,000 from 10,000 people, with the overflow going to the Mozilla Foundation.

The Firefox community is a clear example of how a simple frame, which is good for volunteers and the press, combined with some clever marketing techniques can propel a good free software product from obscurity to fame.

Can KDE and GNOME follow the fox?

Of course Firefox isn't the only free software community working on its marketing strategy. Most of the major projects have been promoted to some extent, be it with a one-off stall at a trade show or OpenOffice.org's impressive forty page strategic marketing plan. So why have they not been as successful, and how can they improve? To answer these questions I will look at KDE and GNOME, the leading free desktop environments, both of which have some marketing activity, and both of which rely largely on their

commercial distributors for promotion beyond the reaches of their respective communities.

Products like KDE and GNOME face several major hurdles: for most computer users they require a migration to a completely new operating system like GNU/Linux or FreeBSD, whereas Firefox can be installed on every major consumer operating system; second, even where users can try them out (on Live CDs, for example), users are presented with a bewildering array of new features requiring a lot of time and interest to digest; and third, where users come into contact with them, the products are bundled with the distributor's own offerings, usually re-branded and slightly modified, meaning that the individual, original products aren't the main subject of promotion. Given these problems, it's apparent that the Firefox community's word of mouth campaign, based on a simple frame and a product that can be tested and promoted with ease, isn't appropriate for all free software projects.

KDE and GNOME face further problems particular to their projects. Marketing is usually a matter of differentiating yourself from your competitors, so in their case you would expect them to regard each other as competition to be beaten. But the projects share code in libraries and standards, they share ideas and regularly inspire each other, and they even share some volunteers. Furthermore, many of their most exciting future features including DBUS, HAL and Cairo are being developed through freedesktop.org, a hub for developing shared, interoperable desktop technologies. They even benefit indirectly from each others' releases, as each improvement makes the GNU/Linux system - which they are packaged as a part of - more attractive. As with many free software projects, they have to emphasise strength through cooperation with competitors.

So what are KDE and GNOME doing at the moment? For the KDE community, promotion work over the past few years has largely focused on trade shows and their news portal, KDE.News (`http://dot.kde.org`). If there has been a marketing strategy so far, it has been the result of volunteers developing common practices rather than any concerted strategic thinking.

Regular contributors from different countries have grouped into national (e.g. KDE GB and KDE NL) or continental (e.g. KDE North America) teams, who attend trade shows to show off their work to geeks and businessmen, armed with a demonstration computer and, usually, some basic literature. Other community members, largely core code and documentation contributors, write for and edit the news web site, providing a regular flow of information on the project's status and the occasional promotional article. These same volunteers tend to write press releases for major versions of the desktop environment, and occasionally informal announcements for other events.

Fig. 2: The KDE promotion team have developed their own press pack for journalists

Fig. 3: The author (second from right) facilitating an interview at aKademy

There aren't many volunteers in the KDE community who devote themselves solely to promotion work, in contrast to Firefox. The Dutch team, KDE NL, has taken the lead in this regard providing a regular team of volunteers at every major event, and producing a range of promotional materi-

als including published articles, entry-level documentation, leaflets and press releases. Other teams are working to catch up, but again volunteer time, resources and a lack of expertise are holding them back. The arrivals of a marketing student and some volunteers with press experience are changing this, with a strategic review of competitors and a comprehensive press pack in the works. But KDE remains a long way off from the sophistication of Spread Firefox.

A big success story for the KDE community was aKademy, its annual conference, held last year in Ludwigsburg, Germany. For the first time, the community had a dedicated press team of three volunteers working full time to promote the event and the project. I was a member of that team, fielding calls from journalists, setting up interviews, writing and sending out press releases, producing a press pack for the event and even writing our own original material for a variety of publications. The event received an unprecedented amount of coverage from the usual tech media and the local media, thanks to the speedy translations of the German member of the team. A small investment of money, for travel costs and the devotion of a week from the volunteers, reaped large rewards for the project.

The situation in the GNOME community is similar. Though they haven't enjoyed KDE's rich tradition of promotion through articles and news items to the same extent, and their annual conference GUADEC (GNOME Users and Developers European Conference) has yet to be promoted on the same scale, they have attended trade shows and courted distributors since their inception. But in recent months a transformation has been taking place on their mailing lists and Wiki. Inspired by a talk at GUADEC, a dedicated team of volunteers have been working on a strategy, developing clearly identified targets (including subsets of users, developers and distributors), materials to reach them with and the beginnings of a set of frames, one for each target market. You can review their efforts on their Wiki (`http://live.gnome.org/MarketingTeam`).

> In general neither communities nor their commercial distributors promote the products directly

The strategies that these two major free software communities are pursuing have met with some success. Not only in terms of winning over new users and developers, and by giving them some bases for evangelising. But also by persuading distributors to use their products and by giving them the raw materials for their own promotional work. However, because of the difficulty in demonstrating a full desktop environment, and consequently, in migrating people to it; because of the difficulty of developing clear frames with targets, tools and incentives; and because of the simple lack of volunteers, they haven't come close to matching the success of the Firefox community.

Conclusion

So where can we go from here? Clearly few free software communities can expect to be as successful as Firefox, but we can still expect worthwhile results. If you're part of a free software community, or if you've got some time to volunteer for this kind of work, the important thing is to think about promotion strategically.

Work out who the likely target markets are, and discuss what strategies you might employ to reach them. For each strategy, think about how much effort you will have to put in, and how far you will move them towards using your product and/or joining your community. Then look at what resources you have and employ them carefully, taking care to avoid wasted time and volunteer burnout (where somebody does too much work for too long). Easy jobs include writing press releases and sending them out, responding to press enquiries, and providing promotional materials like web site buttons and leaflets. If you have the time, attend the relevant trade shows, give presentations and write full articles yourself for trade publications.

Given that community projects aren't selling their product directly, it is a good idea to focus on feature parity and the benefits of free software. Luckily all good free software projects have plenty of raw material that the marketplace will understand. Push freedom from abandonware, your open and transparent development process, freedom from spyware and endless license tracking. Promote sharing and the virtues of free software development methodologies. Demonstrate how free software is usually more secure, cheaper and has fewer bugs. Promote the best features of your product. If you can reduce all of that into a clear, simple set of frames for users, developers, the press and, most important of all, your future promotion volunteers, you'll be

Textbox 1: Marketing strategy checklist

- A simple, clear frame, including ideology and a feature list
- Well-described target markets
- Appropriate materials, targets and incentives for each market
- A press team with contacts and release writers
- A web site to attract volunteers with clear tasks and longer-term jobs

on the road to success.

Free software communities also need to remember the immense power of volunteer recruitment. Marketing is like any other job on a free software project: you need somewhere to start, with immediate or quick feedback and satisfaction. So if you're looking for volunteers, make sure your community has a well-presented, clear list of tasks and jobs so that people can easily participate. This way you will focus effort effectively without having to rope volunteers into jobs they dislike.

There's plenty of potential for guerrilla marketing in the free software world. As people with expertise in media work, marketing and writing come to use free software, so the communities that produce their software stand to gain from their time and expertise. We don't need to, and indeed cannot, rely on commercial distributors to effectively promote free software. The job is left to us, so let's get out there and reclaim the media and marketplace for the community.

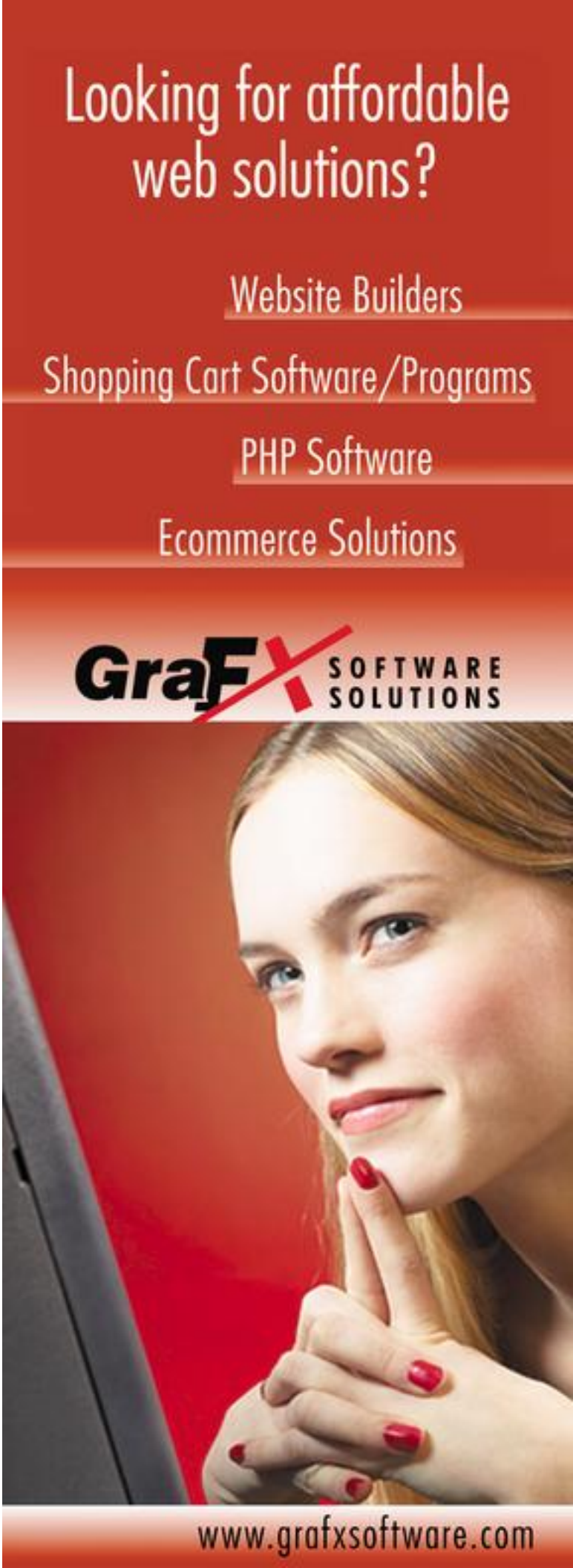
Copyright information

© 2005 by Tom Chance

This article is made available under the "Attribution-NonCommercial" Creative Commons License 2.0 available from http://creativecommons.org/licenses/by-nc/2.0/.

About the author

Tom Chance is a philosophy student, free software advocate and writer. He has worked on promotion with the KDE Project, acting as press officer at their annual conference, aKademy, in 2004. You can contact him through his web site (http://www.tomchance.org.uk)



Yocto^{-24}

OSS Business Solutions

Attention: Senior and IT executives. Gain freedom and save thousands of dollars!

Eliminate proprietary solutions by implementing an open source software strategy!

Receive a **FREE** open source knowledge starter pack containing everything you need to know about open source software, its benefits and how to make it work in your business.

Contact us **before 1 April 2005** and also receive a FREE open source assessment and a consultation session **worth $500** from Yocto's best open source software guru (Gauteng, RSA Only) with no obligation at all.

Yocto OSS has the knowledge, experience, and the skills you require to implement such a strategy. Yocto OSS partners with major players in open source software and is actively involved in the open source software research community.

Contact us via our website http://www.yocto.co.za or email marketing@yocto.co.za with your contact details and the reference number FSM001 to receive your FREE open source knowledge starter pack.

Make use of open source software to fully support your business strategy.

Do it without any risks to your business!

A law for free software

Don’t we have enough laws already?

Maureen O’Sullivan

Free software, also known as open source, libre software, FOSS, FLOSS and even LOSS, relies on traditional software legal protection, with a twist. Semantics aside (I will describe all the above as “free software”), the tradition at law is that free software is copyrighted, like most other software, and is not released, unbridled, to the public domain. Authorial or ownership rights can be asserted as with any bit of proprietary software.

Whilst no solution should be seen as a panacea for all possible ills, the legal lacunae under the present copyright-cum-licensing regime need to be addressed without delay

The twist emerges when one examines what a free software licence actually does in contrast with what its proprietary counterpart seeks to achieve. Whereas the latter aims to control and inhibit the rights normally granted under copyright, such as fair use, the former augments the rights generally available within copyright law. Thus while all licences grant you access to the source code of the program in question, some may allow you to privatise your modifications while others prevent you from exercising this option. This article contrasts proprietary and free software licences in terms of their effectiveness and also gives a brief overview of free software licensing, highlighting legal vulnerabilities and their possible remedies by an intervention of the legislature.

Whilst no solution should be seen as a panacea for all possible ills, the legal lacunae under the present copyright-cum-licensing regime need to be addressed without delay.

Contrasting proprietary and free software licences

An interesting starting point for the purposes of the following discussion is the contrast between the observance rates of proprietary and free software licences. On the proprietary software side, a multitude of forces are amassed against would-be, so-called software “pirates”, which probably tarnishes a large portion of the population in general. We have copyright law - an old bottle, arguably not well tailored to house this new wine - but chosen in the 1980s as an appropriate legal protection for a number of reasons, including the fact that it was considered a good idea to treat software code in the same way as you would treat a literary work. Signatories to the Berne Convention could provide this protection almost instantaneously and, therefore, a rapid, international coverage for software was achieved.

Copyright is not the same in all countries so software owners get a bundle of rights rather than a single one that will suffice in every jurisdiction. Moreover, proprietary software licences tend to restrict these rights, in return for permission to use the work in a limited way. Copy protection is also used in tandem with copyright law and this creates a

physical fence to keep software "thieves" and other miscreants out. Despite the copyright bargain of leaving ideas free while protecting expression, patent law has subsequently been applied to this material, which was supposed to lie in the public domain. Industry representative bodies, such as the Business Software Alliance, are fond of pointing out the humungous losses caused by "pirates" and they have developed a moral arsenal of a type of online confessional by proxy where disgruntled employees are encouraged to report their errant bosses.

Despite the copyright bargain of leaving ideas free while protecting expression, patent law has subsequently been applied to this material, which was supposed to lie in the public domain

Often the threat of a lawsuit, whether well founded or vexatious, is enough to send an institution into a flurry of panic about a visit from BSA representatives. None of these increasingly restrictive and repressive practices appear to have much effect - in many parts of the world, piracy rates still soar and in other parts, governments, tired of continual name-calling by foreign bodies, have decided to abandon proprietary software and replace it with free software. This apparent panacea has not proven to be an acceptable solution for the proprietary software folk either. Their motto would appear to be "my product, my way".

By way of contrast, free software licences are rarely litigated and the most prominent, the Free Software Foundation's GNU GPL, has only recently made its court debut, which in its 15 year history, is a remarkable achievement. Free software licences enshrine a sort of customary law or act as a declaration of customs within hackerdom. The preamble of the GNU GPL, in particular, employs a style of language richly reminiscent of the often countered "We the People…" sections from the constitutions of many nations. Its terms lay out clearly what one may and may not do with the code to which the licence applies. It is a sort of lex scripta or transcription of long-established and practised customs. So when a failure to observe such norms is encountered, the mail in the mailbox of the guilty party will steadily increase into a stream of less and less friendly messages, the longer the non-observance of the licence persists.

The FSF is a veteran of this informal style of enforcement and, in contrast with the BSA, its moral prescriptions largely keep its flock in line. Participants in the free software community generally have an incentive to comply as there are ongoing relationships and reputations at stake, which could be affected, should a prolonged violation be sustained. It is an insider's law and is inherently democratic because it is chosen from a range of different options and voluntary observance is the norm. Outsiders who have benefited from free software also generally contribute back to the free software commons - IBM being a case in point - and they respect established licences, as a failure to do so would provoke hostility and ultimately damage their business.

Free software's legal vulnerability arises from businesses that do not participate in using and contributing to the free software commons, and whose market hold is threatened by the incursion of free software into their monopoly. Those who in their day relied on a sympathetic legal system to convert software sharing into theft may also depend on the same system, 25 years on, as a way to dampen free software's growing popularity in business and government software around the world. The next section delves a bit deeper into whence these vulnerabilities may arise.

Free software's legal threats

Free software also depends upon the copyright system, but in an amusing and subversive way. Had legislators done the right thing in the early 1980s, they would have created a sui generis law, specially tailored toward enhancing this resource and not facilitating the creation of monopolies in the operating systems market. However, this inappropriate law was a fait accomplit and has been used by free software advocates to achieve the opposite of what it was intended to achieve.

Thus free software "law" has operated up to now as a community custom rather than as a legally binding document

Thus free software "law" has operated up to now as a community custom rather than as a legally binding document. Its vulnerabilities stem from the fact that what may be, in one country copyrightable, is not necessarily in another.

This also applies to the licence which sets out the terms on which the copyrighted work may be used, because contract or licensing laws differ in each country too. Moreover, at times, particularly in common law jurisdictions, judges may have difficulty in determining ownership rights in collaborative works. The FSF encourages their programmers to assign their code to the Foundation, whereas Linus Torvalds now requires the author of each chunk of code to be identified. In other cases, there may be a delay in choosing a licence; or a licence may not be passed on to third parties, who would then not be bound by any conditions stated in the licence.

> Free software's legal vulnerability arises from businesses that do not participate in using and contributing to the free software commons, and whose market hold is threatened by the incursion of free software into their monopoly

Copyright provides a fall-back position in such cases, as it does bind third parties. However, with the increasing internationalization of code creation and the growth of the internet, pursuing violators and resolving matters in favor of the authors of free software may not be so straightforward. Effectively, at present, free software relies on a patchwork of various laws, which may or may not provide the desired support at crucial moments. An ideal would be an international, harmonized piece of legislation. This would bind third parties, because a contract would no longer be necessary to set out the agreed terms. A statute could do this and violators would have no place to hide if they failed to observe the terms of a "Free Software Act". Any such legislation should not be imposed from the top down but, rather, should emanate from the ground up. It should be drafted as a collaborative venture between lawyers, technicians, politicians and also, most importantly, free software creators, without whose enormous efforts, you would not be reading this article because I would not have written it.

The Free Software Act project

I concluded several years ago that free software licenses, and particularly the GNU GPL (possibly because this is the most demanding licence in terms of what it expects back from users of its software), had become a sort of customary, international law within hacker communities.

> I concluded several years ago that free software licenses, and particularly the GNU GPL had become a sort of customary, international law within hacker communities

Despite the fact that it was designed in a common law jurisdiction, countries from the civil law tradition have adopted it without too much further thought. It binds communities more through a sense of obligation rather than a fear of an external law. When democracies function well, laws should be enacted to protect commonly developed resources from outside threats; and governments should work with the people and not as puppets of large corporations, whose interests tend to clash with those of the masses. Law can often emanate from the ground for a plethora of reasons but in this case in particular, a recognition of hacker-law needs to take place by legislatures and courts around the world. Free software has matured and the law needs to keep up with it, not to thwart it, not to control or curtail it. It should help cultivate its development in the manner with which its creators would deem appropriate. These creators, of course, will not hold a unanimous opinion. Therefore, it is imperative that lawmakers work with the community as a whole and while in the process of drafting legislation in this area, they should be led by their vision.

> Copyright is not the same in all countries so software owners get a bundle of rights rather than a single one that will suffice in every jurisdiction

The importance of this cannot be overstated: free software is becoming crucial to the development of many societies and, in certain cases, cannot actually be replaced by proprietary software because it has become more than just a means to contact people or find out information.
It is now deeply embedded in education, society and local government, the best example in the world of this being in

Fig. 1: The Free Software Consortium's home page

the region of Extremadura in Spain.

About 15 months ago, I sketched a rough draft of how I thought such a law should look and called the draft the Free Software Act. The Free Software Consortium and I have worked hard to promote this and I got as far as version 3, all of which can be viewed on our web site (`http://www.fsc.cc`).

Free software is becoming crucial to the development of many societies and, in certain cases, cannot actually be replaced by proprietary software because it has become more than just a means to contact people or find out information.

I am just putting the finishing touches on version 4 which will be available there shortly. The difference between version 4 and the earlier versions is that I have now set up an international, interdisciplinary committee to advise on its terms and to promote its enactment around the world. It addresses the aforementioned legal lacunae, which currently leave free software vulnerable to attack by hostile forces. An ideal would be for such a law to be passed at an international legal forum but if this is not possible, it could also be used at national level, local level or even as a political declaration to focus attention on what is at stake. As this project is intended to be representative, democratic, interdisciplinary and grassroots, comments are always welcome either at this web site (`http://www.fsc.cc`) or direct to me at maureen AT fsc DOT cc.

Conclusion

This article has sought to draw attention to why legally enforced proprietary software licences are treated with such disdain by the people to whom they are meant to apply. And, by way of contrast, how free software licences are rarely litigated, as their enforcement doesn't rely on an arsenal of hostile forces and propaganda but, rather, negotiation, albeit sometimes tough. It traces the manner in which free software licences rely on existing law for their operation. Further, it discusses how, as the market for free software expands and the numbers and associations of the community grow, the licences may lose some of the force they have had up to now because external actors may not feel the same obligations as community members.

Therefore, it is imperative to enshrine their customs in statute, directive and/or international law so that however and wherever a custom is broken, it can be enforced. Failing to act now, while free software is still relatively unthreatened, is a risky strategy. At this point in time we should be building legal defences to ensure that a repeat of the privatisation of software a quarter of a century ago is not repeated.

Copyright information

© 2005 by Maureen O'Sullivan
Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Maureen O' Sullivan wrote a thesis in socio-legal aspects of Linux and is now a researcher and lecturer in international software law, covering both licensing and legislation. She has published extensively in the field and has recently undertaken a number of consultancies in the area, both in Europe and Latin America.

The Libre Culture Manifesto

A manifesto for free/libre culture

David M. Berry Giles Moss

We have written this manifesto always wishing to unfold the concept and practice of free/libre and open-source. We wanted it to stretch out so that it might take us in new directions. To start off with, we were sure that the practice of non-proprietary software code production was not a narrowly technical or economic affair, but something that was always also socio-political. Employing a critical political economy framework, we wanted to draw out the socio-political aspects of free/libre and open-source in an age of "creative capitalism" and "creative industries", where the exploitation of concepts and ideas through intellectual property (supported by new prescriptive technologies) has become so important to profit.

A constellation of interests is now seeking to increase its ownership and control of creativity... But this is a disaster for creativity, whose health depends on an ongoing, free and open conversation between ideas from the past and the present

At the same time, the copyleft ethos was already stretching out before us in myriad ways. In those places where creativity was being divided up and exploited by private interests for profit (viz., not just software, but also art, music, writing, science, design and so on), an ethos of sharing concepts and ideas was widening in response. It is stirring for us that the concept and practice of collective creativity continues to deepen in this way. We just hope it does not fold up into itself, as some members of the movement may wish, but that it continues to recognize its current socio-political significance, and that it stretches itself out in new creative alliances that simultaneously confront and transform the present...
(DMB & GM - Nov 2004)

The Libre Culture Manifesto

A constellation of interests is now seeking to increase its ownership and control of creativity. We are told that these interests require new laws and rights that will allow them to control concepts and ideas and protect them from exploitation. They say that this will enrich our lives, create new products and safeguard the possibility of future prosperity. But this is a disaster for creativity, whose health depends on an ongoing, free and open conversation between ideas from the past and the present.

In response, we wish to defend the idea of a creative field of concepts and ideas that are free from ownership.

1

Profit has a new object of affection. Indeed, profiteers now shamelessly proclaim to be the true friend of creativity and the creative. Everywhere, they declare, "We support and

protect concepts and ideas. Creativity is our business and it is safe in our hands. We are the true friends of creativity!"

2

Not content with declarations of friendship, profiteers are eager to put into practice their fondness for creativity as well. Action speaks louder than words in capitalist culture. To display their affection, profiteers use legal mechanisms, namely intellectual property law, to watch over concepts and ideas and to protect them from those who seek to misuse them. While we are dead to the world at night, they are busily stockpiling intellectual property at an astonishing rate. More and more, the creative sphere is being brought under their exclusive control.

An artificial scarcity of concepts and ideas can then be established. Much money is to be made when creative flows of knowledge and ideas become scarce products or commodities that can be traded in the market place

3

The fact that the profiteers are now so protective of creativity, jealously seeking to control concepts and ideas, ought to rouse suspicion. While they may claim to be the true friends of creativity, we know that friendship is not the same as dependency. It is very different to say, "I am your true friend because I need you", than to say, "I need you because I am your true friend". But how are we to settle this issue? How do we distinguish the true friend from the false? In any relationship between friends we should ask, "Are both partners mutually benefiting?"

4

The profiteers' insatiable thirst for profit clearly benefits from their new friendship with creativity and the creative. Unlike physical objects, concepts and ideas can be shared, copied and reused without diminishment. No matter how many people use and interpret a particular concept, nobody else's use of that concept is surrendered or reduced. But through the use of intellectual property law – in the form of patents, trademarks and particularly copyright – concepts and ideas can be transformed into commodities that are privately regulated and owned.

An artificial scarcity of concepts and ideas can then be established. Much money is to be made when creative flows of knowledge and ideas become scarce products or commodities that can be traded in the market place. And, increasingly, intellectual property law is providing profiteers with vast accumulations of wealth.

5

Informational, affective and knowledge-based labour has now become a central driver of profit. Indeed, immaterial labour is increasingly replacing industrial manufacture as the main producer of wealth in the age of technological capitalism. With these developments in the productive processes, a new embodiment of profit emerges. Alongside the landlords that controlled agriculture and the capitalist factory owners that controlled manufacture, vectors— the owners of the distribution, access and exploitation of creative works through valorisation— have emerged. It is these same vectorialists, of course, that are now so vocal in their claim to be the true friends of creativity and the creative.

6

For many of us, the thought of intellectual property law still evokes romantic apparitions of a solitary artist or writer seeking to safeguard her or his creative work. It is therefore unsurprising that we tend to view intellectual property law as something that defends the rights and interests of the creative. Perhaps, in some removed and distant time, there was a modest respect in this notion. But this romantic vision of the creative is certainly ill at ease with the current capitalist reality.

7

The world in which creative people now find themselves is a social factory or a society - factory (Virno, P. & Hardt, M; 1996). The vectors view the whole social world of creativity

and creative works as raw material for commodification and profit. Creative people have thus become de facto employees of the vectors, if not their actual ones. Each concept and idea they produce is available to be appropriated and owned by the vectors through the use of intellectual property law. What is more, the vectors continually lobby to extend the control of these laws for greater and greater lengths of time. Because the vectors have now made intellectual property law their own, we can from now, more accurately, term these laws, "vectoral laws".

8

The creative multitude is becoming legally excluded from using and reinterpreting the concepts and ideas that they collectively produce. In addition, this legal exclusion is being supported by technological means. Using technology as their delegates, the vectors seek to enforce vectoral law by instantiating their interests within the technical code that configures information, communications, networks and devices. To do so, they are currently developing and configuring ever more closed technologies and disciplinary machines.

Digital rights management software, for example, sequesters and locks creative works, preventing their copying, modification and reuse. The vectors can by using these prescriptive technologies deny access to those who cannot pay or to those whose sympathises and support are not assured. They can also exclusively determine how ideas and ±! concepts are to be used in the future. In the current era of technological capitalism, public pathways for the free flow of concepts and ideas and the movement of creativity and the creative are being steadily eroded — the freedom to use and re-interpret creative work is being restricted through legally based but technologically enforced enclosures.

9

This development is an absolute disaster for creativity, whose health depends on a free and ongoing conversation and confrontation between concepts and ideas from the past and present. It is shameful that the creative multitude is being excluded from using the concepts and ideas that they collectively produce. Creativity is never solely the product of a single creator or individuated genius. It always owes debts to the inspiration and previous work of others, whether these are thinkers, artists, scientists, paramours, listeners, machines or friends. Creativity, as a fusion point of these singularities, cannot subsist in a social nothingness. Concepts and ideas depend upon their social life — *and it could not be otherwise*.

Digital rights management software, for example, sequesters and locks creative works, preventing their copying, modification and reuse

10

An analogy can be drawn with everyday language: that is, the system of signs, symbols, gestures and meanings used in communicative understanding. Spoken language is shared between us. A meaningful utterance is only made possible by drawing on the words that freely circulate within a linguistic community of speakers and listeners. Language, then, is necessarily non-owned and free. But imagine a devastating situation where this was no longer the case. George Orwell's depiction of a 1984 dystopia — and the violence done here to freethinking through newspeak — helps to illustrate this. In a similar way, the control and ownership of concepts and ideas is a grave threat to creative imagination and thought, and so also a danger to what we affectionately call our freedom and self-expression.

11

Until recently, the creative multitude could decide either to conform or rebel. In conforming they became creatively inert, unable to create new synergies and ideas, mere producers and consumers of the standardised commodities that increasingly saturate cultural life. In rebelling, they continued to use concepts and ideas in spite of vectoral law. Labelled "pirates", "property thieves" and even "terrorists", they were then answerable as criminals to the courts of global state power. In other words, a permanent state of exception, a political emergency, was declared, which, together with the disciplinary norms of a propertised control society, was then used to justify and extend the coercive use

of state power and repression against an increasingly criminalised culture of creativity.

But as we will soon discuss, a growing number of the creative have now moved beyond both conformity and rebellion, through an active resistance to the present and the creation of an alternative creative field for flows of non-owned concepts and ideas.

12

The vectors and their representatives will make immediate objections to all we have said. The profiteers will turn proselytizers and exclaim, "If there is no private ownership of creativity there will be no incentive to produce!" The suggestion that the ownership of knowledge and ideas promotes creativity is a shameful one, however plausible it may seem from the myopic perspective of profit. To say that creativity can thrive while the creative lack the freedom to reuse concepts and ideas is clearly upside-down. After giggling a little at this, we should now turn this thinking the right way up.

13

According to this "incentive" claim, there cannot have been any creativity (i.e., art, music, literature, design and technology) before the ownership and control of our concepts and ideas. This seems like fantasy. Historians frequently profess to us that creativity was alive and well in pre-capitalist times, before the advent of intellectual property laws. But even so, we might concede that history is now enough of a fiction to raise some doubt about the form of previous incarnations of creativity and the creative. The incentive claim, however, is even more risible when it implies that there cannot be any creativity currently operating outside of the vectoral property regime. This of course contradicts our current experiences as historical actors and witnesses. We can now be sure of something that we have always already known — creativity is irreducible to the exploitation of intellectual property.

14

A new global movement of networked groups that operate across a variety of creative media (e.g., music, art, design and software) is now emerging. These groups produce a *gathering* (Versammlung Heidegger; 1951). of concepts, ideas and art that exist outside the current vector property regime. The creative works of the Free/Libre and Open Source communities, for instance, can all be freely examined, challenged and modified.

> Whereas copyright operates through law to prevent the modification and re-use of concepts and ideas, copyleft ensures that these concepts and ideas remain openly available and not capable of being privatised

Here, knowledge and ideas are shared, contested and reinterpreted among the creative as a community of friends. The concepts and ideas of these groups, like the symbols and signs of language, are public and non-owned. Against the machinations of profit, these groups are in the process of constituting a real alternative — of constructing a model of creative life that reflects the force and desire of the creative multitude.

15

Through the principles of attribution and share-alike, existing works and ideas are given recognition in these communities. This means that while creative work may always be copied, modified and synthesised into new works, previous creative work is valued and recognised by the community for its contribution to creativity as a whole (and rightly so). Attribution and share-alike are constitutive principles of the Free/Libre and Open Source movements, and chromosomes of the new mode of creative life that their social practice intimates.

16

These movements adopt an ingenious viral device, implemented through public licences, known as copyleft.

This ensures that concepts and ideas are non-owned, while guaranteeing that future synergies based on these concepts

and ideas are equally open for others to use. Whereas copyright operates through law to prevent the modification and re-use of concepts and ideas, copyleft ensures that these concepts and ideas remain openly available and not capable of being privatised. In this way, copyright ("all rights reserved") is stood back on its feet by copyleft ("all rights reversed"). It now stands the right way up for creativity and can once again look it in the eyes.

17

More broadly, we can say that non-owned creative works are created by singularities formed into machines of struggle (e.g, GNU, bit-torrent, nettime.org, autonomedia, SchNews, the Zapistas, Linux, Indymedia, Loca Records). These are horizontal and decentred molecular networks of actors, both human and non-human. These can and should be differentiated from the more centralised, disciplinary machines to which the concept network is now so liberally applied (e.g., "network firms", "network states", "network wars"). As such, they should also be distinguished from vectoral machines (e.g., capitalist corporations, WTO, IMF, the World Bank), which are closed, hierarchical, proprietary machines that configure and territorialize networks, concepts and ideas.

18

Machines of struggle are continually being enrolled into new alliances and relations. As the vision and practice of non-owned creativity gathers in strength, these rhizomatic arrangements are both deepening and widening. Just as the violence of the vector's disciplinary regime is seeking to intensify, it is being met with a real counter-power. This countervailing force finds its form and strength, not through any individual nucleus or singularity standing alone, but through broader relations and alliances. More accurately, therefore, we are talking here of circuits of counter-power – *machines of struggle in creative alliances.*

19

These circuits of counter-power bring forth the scope for resistance, the capacity for agency and thus the hope and promise of future worlds. When linked together, machines of struggle are able to confront and challenge the vectoral regime as a real force, collectively armed against the territorializing effects of vectoralist capital. Circuits of counter-power provide the conditions and capacity for transformative constitutive action. Such circuits are but one moment of the potential power of the creative multitude as organised and effective transformative agents.

20

We believe that the creative multitude should form themselves into machines of struggle and establish alliances with broader circuits of counter-power. In so doing, they contribute towards the idea and practice of non-owned creativity and the untimely model of creative life that it intimates. Through collective production and shared creative alliance, they will defend and extend creativity against those who shamelessly remain wedded to the language and practice of private property and profit, and who continually attempt to territorialize and configure for the purposes of control and ownership.

21

Indeed, we — who are already quite a crowd — must defend the idea and practice of non-owned creativity. For it is only the creative multitude, when organised and enrolled into circuits of counter-power, who will determine whether a possible transformation of our times is realised. This is a movement that is acting "counter to our time and, let us hope, for the benefit of a possible time to come" (Nietzsche; 1983, p60). *Creativity is creating resistance to the present.*

Bibliography

[1] Adorno, T. & Horkheimer, M. (1976) The Dialectic of Enlightenment

[2] Benjamin, W. (1935) The Work of Art in the Age of Mechanical Reproduction

[3] Deleuze, G, & Guattari, F. (1999) What is Philosophy?

[4] Deleuze, G, & Guattari, F. (2001) A Thousand Plateaus

[5] Foucault, M. (1990) The History of Sexuality, Vols 1, 2 & 3

[6] Hardt, M. & Negri, A. (2000) Empire

[7] Feenberg, A. (1991) Critical Theory of Technology

[8] Heidegger, M. (1993) “Building Dwelling Thinking”. In Basic Writings.

[9] Machiavelli, N. (1958) The Prince

[10] Latour, B. We Have Never Been Modern

[11] Martin, B. (2003) Against Intellectual Property (http://danny.oz.au/free-software/advocacy/against_IP.html)

[12] Marx, K. (1974) Capital

[13] Marx, K. (1974) Grundrisse

[14] Marx, K. & Engels, F. (1974) The German Ideology

[15] Nietzsche, F. (1983) Untimely Meditations

[16] Rose, N. (1999) Powers of Freedom

[17] Schmidt, C. (1995) The Concept of the Political

[18] Stallman, R. (2002) Free Software, Free Society

[19] Virno, P. & Hardt, M. (1996). Radical Italian Thought.

Copyright information

© 2005 by David M. Berry, Giles Moss

This article is made available under the “Attribution-Share-alike” Creative Commons License 2.0 available from http://creativecommons.org/licenses/by-sa/2.0/.

About the author

David is a researcher at the University of Sussex, UK and a member of the research collective The Libre Society. He writes on issues surrounding intellectual property, immaterial labour, politics, open-source and copyleft. Further information can be found at http://www.libresociety.org. **Giles** is a doctoral candidate at New College, Oxford researching e-government and the sociology of translation. He is also a member of the Libre Society research collective.

A PLANT NEEDS WATER TO GROW
FREE SOFTWARE MAGAZINE
NEEDS SUBSCRIBERS TO PROSPER!
Free Software
MAGAZINE
BY SUBSCRIBING YOU WILL BE SUPPORTING A MAGAZINE WHICH BELIEVES IN FREE SOFTWARE. ALL OUR ARTICLES ARE RELEASED UNDER THE GNU FREE DOCUMENTATION LICENSE, ENHANCING EXISTING INFORMATION ON FREE SOFTWARE.
SUBSCRIBE NOW!
HTTP://WWW.FREESOFTWAREMAGAZINE.COM/SUBSCRIBE

Richard Stallman's blog

Selected entries from Richard's blog (http://agia.fsf.org/rms-blog), from November 2004 to December 2004

Richard Stallman

Venezuela (November 15, 2004 to November 22, 2004)

I spent a week in Venezuela, giving a speech and some interviews at an event which invited speakers from all across Latin America. During the event, the state oil company PDVSA announced its decision to switch 100% to free software. Their decision is not based on convenience or cost; it is based on sovereignty.

Their computers used to be handled by a US company, SAIC. When opponents of President Chavez tried to drive him from office by shutting down oil protection, the US government helped out by telling SAIC to prevent them from using their computers. PDVSA therefore knows from experience that using non-free software means you are at the mercy of the developers, and has decided to solve the problem for good and all.

I was supposed to be interviewed on a breakfast TV show, and had to wake up at 0530 for it. When we arrived at the station, we found out that everything had been pre-empted; the prosecutor in charge of cases against people who participated in the Bush-sponsored attempt to overthrow Chavez had been killed with a bomb in his car. This act of terrorism was most likely carried out by some of the same people that Bush supports. But there are no plans to respond by abolishing civil liberties, as has been done in the US.

On Sunday I went for lunch with Sergeant Torres, who has converted many of the Venezuelan Army's servers to GNU/Linux. He brought his wife and son; his son is something like 8 years old and already starting to use free software.

Fig. 1: Avila national park in Venezuela

For dinner I went with some Venezuelan free software activists to a restaurant near the top of the mountain ridge that separates Caracas from the Caribbean. We couldn't see the sea as we sat down to eat, around 540pm, because clouds were in the way. The appetizer, a soup, was marvelous but the rest of the dinner was not quite as good.

After dinner we went to sit at a table just outside the restaurant to have coffee or tea. I felt like doing something else, so I played a Bulgarian tune on the recorder. A person sitting at another table nearby said, "That sounds Celtic, or Breton."

It is very common for people who are not accustomed to

Balkan folk dance music to think it sounds Celtic, but few say something so specific as "Breton." So I played a Breton dance tune and said, "That was a Breton dance, Kost-ar-c'hoed." Then I realized the man was speaking Spanish with an unusual accent, so I asked him, "Etes-vous français?" He said that he was, and in fact from Brittany. We got into a conversation and I explained free software to him and the man with him, who turned out to be the restaurant's owner. Since he was interested, my friends then joined the conversation. Meanwhile, by this time the clouds had dissipated and we could see the shore and the sean, beautiful 6000 feet below.

Colombia (November 22, 2004 to November 24, 2004)

The following day I went to Colombia. The strength of the free software community there really surprised me. I met with people from several user groups, and on Tuesday met with enthusiastic representatives of Colombiás major universities as well as the Mayor of Bogota, to whom I suggested that the most important way to support free software was to switch to it in the city's schools. We agreed they would have a plan ready when I return in March.

Venezuela (December 1, 2004 to December 6, 2004)

I returned to Venezuela after Mexico, for a conference called Artists and Intellectuals in Defense of Humanity. On Saturday, our main activity was a meeting where President Chavez would speak. I had lunch that day with Sergeant-Major Torres, who has converted the Venezuelan Army's servers to GNU/Linux, and his wife.

Since we've become friends, I encouraged him to come back to the hotel and try to get into the meeting too, figuring the security would probably ok his entry, and they did.

The first speaker was Sr. Perez Equivel, a Nobel Peace Prize winner. He told a fable of a chef who called together various animals to ask each one, "What sauce would you like to be cooked with?" The chicken responded, "Actually I'd rather not be cooked at all", but the chef said, "That's outside the discussion - all you can decide is which sauce."

An invited American speaker then compared the invasions of Afghanistan and Iraq to the wars of aggression for which the Nazi leaders were convicted. (I supported the invasion of Afghanistan because of the totalitarian cruelty of the Taliban, which goes beyond Bush.) He described Fallujah as the Guernica of the 21st century, and warned that all countries now feel fear of being attacked by the new empire of the US. He referred to the celebration of torture, in Guantanamo as well as in Abu Ghraib, as a brutal threat against the whole world. Regime change imposed by the US, over and over, has led to a regime much worse than what preceded it; he cited the Congo, with its 37 years of tyranny, and the US-organized coup in Haiti this year, and accused the US of being behind the attempted coup in Venezuela in 2002. (The US recognized the coup leaders' government immediately.) But the US is not merely a global scofflaw; Bush intends to be above the law, as shown by the US plans to exempt itself from the International Criminal Court - feeble as that is.

His remarks in English were interspersed with Spanish translations that were often absurdly and ridiculously incorrect. The interpreter seemed to translate most of the concepts that had just been mentioned, but often garbled their relationship. Most amazingly, she translated Martin Luther King's famous words as "tengo sueño" (I feel sleepy) instead of "tengo un sueño" (I have a dream). I can't imagine how a native Spanish speaker could make that mistake.

Then Daniel Ortega, former president of Nicaragua, entered the hall and people applauded him.

A deputy (congressman, more or less) from Argentina then spoke, and referred to the problem of the dictatorship of the unidimensional corporate media. He proposed setting up a Latin-American alternative to CNN which would break its dominance over news. He then referred to the bombing of Hiroshima as an act of terror - a somewhat controversial idea to an American, but possibly justified.

Then President Chavez spoke. He covered many topics, and I wished I had asked for a simultaneous translation receiver, because I couldn't always follow his Spanish.

Chavez is greatly admired by Venezuelans, who praise him in more glowing terms than I would praise anyone (even the people I most admire). But Chavez doesn't ask people to praise him - he directs people's admiration to Bolivar, San Martin, and various others who fought against the colonial rule of Spain. He refers to them often, not just to their names but also to their writings and their deeds, in a way that Americans used to talk about Washington, but probably

with greater accuracy. The national anthem, which I think was changed under Chavez, praises the "brave people".

He spoke about the recent first graduation of Project Robinson 2, a project for teaching literacy and basic school knowledge to adults who didn't have a chance to learn before. (This project is named after Bolivar's teacher.) He hopes to extend this program to over a million people, to help them escape from poverty. He spoke of the kitchens established for poor people, including street children, and how they have been situated near to Project Robinson 2 so as to lead them to get an education also, so they could "be their own liberators". He spoke of educational grants of $100 a month that have been given to half a million people so they can study. He responded to criticism that this money should be spent on building "infrastructure", saying that building a sovereign people with dignity is more important than constructing buildings or highways. If my memory is right (I'm not sure of this), I think he also read a description from an old book of some of Bolivar's educational policies.

Chavez spoke about the 70 clinics in poor neighborhoods that have started operating in the past year; the plan is to expand to 1400 of them, to cover 95% of the population. I visited one of these clinics, in the city of Barinas, while other invitees went to other cities and saw clinics operating there. In these clinics, about half the doctors are Cuban and half are Venezuelan. Cuba has an excellent medical system, aside from the shortage of medicines due to the US embargo, and trains many doctors; meanwhile, people told me that many Venezuelan doctors didn't want to get their hands dirty with poor people. Chavez didn't try to force them, he just hired Cuban doctors.

Venezuela has many kinds of cooperation with Cuba, which makes sense since the US tries to attack them both. However, I am puzzled that Chavez, who both speaks highly of democracy and practices it scrupulously, also speaks so highly of Cuba. He turns a blind eye to the fact that Castro does not hold elections, and has imprisoned a number of people for criticizing the government.

Chavez quoted the results of a survey called Latinobarometro, which found that in most countries in Latin America, the number of people who believe that democracy is the best form of government has fallen since 1996 - often fallen by 20% or more. In Venezuela, however, the support for democracy increased by 12% over that period. Venezuela has among the smallest percentage of people who say they might prefer a military government, and likewise for the percentage who say they would give up freedom for the sake of order. He drew this conclusion: support for democracy is falling in many countries because their democracies have failed to deliver what the people want. They have let the IMF, the megacorporations, and/or a national oligarchy exercise power.

Early in his speech, Chavez said that he wanted especially to hear our criticisms of his policies. He later showed this was not just idle talk, because after speaking for perhaps 40 minutes he asked for questions from the audience. There were a few hundred of us in the auditorium, and quite a number wanted to ask questions or give their messages of support, so we were asked to limit ourselves to 3 minutes each. Chavez spent over 4 hours listening and responding. After listening to a number of questions, and (apparently) taking notes, he responded to them one by one.

Could you imagine any recent president of the US holding such a meeting? Bush doesn't even accept unfiltered questions in press conferences.

In one answer, Chavez explained how Venezuela practiced voter suppression in the past. For instance, poor people were give only provisional identity cards, so they were unable to vote. He explained how he had sent trucks with computers and printers to give poor people real identity cards so they could vote, and told how a child (who I suppose wasn't old enough to vote) was happy that he now had a real identity card with a picture of Bolivar on it.

Given my persistent opposition to national ID cards, I feel somewhat less than unalloyed happiness about this solution to the problem, although it was a good thing to make sure poor people could finally vote.

Chavez remarked that the term "democracia popular" ought to be redundant, because "demo" means "the people"; but democracy has degenerated so far in most countries as to make the term meaningful.

Another answer that he gave, when asked about the persistent US intervention in Colombia, is that in the past decade or two some had claimed imperialism had become benevolent, like a protecting father; but that after examples like Colombia and Iraq, this theory had become incredible.

He was asked what he thought about the "war on terrorism", and said that it is "more terrorism". "The only way to end terrorism is with justice." I applauded that (as I did some other things). He condemned the world's armament

spending, above all in the US, for the waste of not using this money to reduce poverty.

When it was my turn at the microphone, I raised the issue of the new law to regulate the broadcast media. This law has been cited by the opposition as a form of tyranny. I obtained a copy of the text a couple of weeks ago, and found no tyranny, but a number of rules that were stated rather broadly, and whose natural interpretations could prohibit broadcasting things from Harry Potter to various somewhat dangerous sports to my own opinions about unauthorized copying. (The term "apología al delito" seems to include that.) The law had been in discussion for over a year, but in an agonizing coincidence, just after I had written down comments on the text and started to circulate them, the law was voted on. If only the timing had been a little different, I might have been able to help make it better.

I explained how what I say in my speeches appeared to be prohibited by this law, and said that I was asking, not in the spirit of opposition but in a spirit of friendly criticism, to make sure this law would not restrict the expression of opinions on radio and TV.

Chavez responded at length to my question, as he did to many questions. Unfortunately, I couldn't follow all of his response clearly.

During the event, the state oil company PDVSA announced its decision to switch 100% to free software. Their decision is not based on convenience or cost; it is based on sovereignty

Chavez explained certain distinctions, about who would be legally responsible for certain kinds of broadcasts, using the example of an interview with a person who said he wanted to kill the president, and whether the interviewer actively contributed to expressing that sentiment. This example is not unrealistic in Venezuela; he told how the TV stations had, for a period of weeks, constantly called on the public to attack Chavez (but the public didn't). The opposition now is no longer so directly violent, but still harsh by US standards. In the airport, as I left Caracas, I looked at an opposition newspaper; it condemned the government for how it has honored the recently slain prosecutor, who was prosecuting participants in the attempted coup. You would never see a major US daily newspaper criticize Bush policies so harshly.

However, choosing that example missed the point somewhat, since calling for violence is not the same kind of case as the one I raised. Perhaps in my question I should have explicitly distinguished expressing opinions from fomenting violence. I felt I had had just one chance, and I had fumbled it just a little, and wasted the chance. I felt like a failure.

About 20 minutes later, as the meeting ended, I asked Sergeant Torres and others to explain to me the parts of Chavez' answer that I had not entirely heard or understood. They told me he said that the constitution, which overrides any law, guarantees the right to express opinions, and also that the requirements in this law will not be enforced by judges but rather by a "media responsibility committee" made up of representatives of the media, civil society, and government, which would apply all the requirements in a less strict way. People also said that the word "apología" is understood in Venezuelan law to mean something beyond merely expressing an opinion; only a direct call to commit a crime would qualify as "apología al delito".

As a result of that, I feel my concern has been mostly resolved, but I'd still feel more comfortable if the law said this more explicitly. So I still feel a burning sense of futility from having come to the issue just barely too late.

On Sunday evening I participated in a dinner which was concerned with the plans to start Canal Sur, as they call the planned alternative to CNN. Many suggested making it a "revolutionary TV" station, and following the best progressive ideas in all ways - for instance, one suggestion was to try to represent not just Latin America but all downtrodden parts of the world.

I suggested, rather, that they follow the model of Al Jazeera, which doesn't call itself revolutionary and doesn't try to be progressive in all ways, but has succeeded in showing millions what Bush doesn't want them to see.

Spain (Santander and Bilbao) (December 13, 2004 to December 16, 2004)

My short trip this month began with speeches in Santander and Bilbao, Spain. Traveling to Santander gave me a chance to see the beautiful rock-surrounded beaches; we also had time for a brief trip to the mountains of Asturias to the west.

Fig. 2: A beach near Santander in Spain

In several languages it told me that the record could be played on Windows and Mac systems. It didn't say that the record can't be played on GNU/Linux systems with free software, but it wasn't hard to deduce that implication. They had given me a Corrupt Disk as a gift!

After my speech in Bilbao, part of an event sponsored by the Basque region's government to promote free software in local businesses, my hosts gave me a bag with a few gifts - designed, I would guess, to introduce the visiting foreigner to something about the region (mainly with a view to drumming up business, I think). The next morning, as I was getting ready to leave, I unpacked the bag. One of the gifts was a record of music by Kepa Junkera, a well-known Basque musician. I had heard the name but never heard his music, and I wanted to listen. But the front of the box said "Copy Control". Suspicious, I read the back of the box. In several languages it told me that the record could be played on Windows and Mac systems. It didn't say that the record can't be played on GNU/Linux systems with free software, but it wasn't hard to deduce that implication. They had given me a Corrupt Disk as a gift!

I think this was the first time I actually held one in my hands. How ironic. I had mentioned the EUCD in my speech, and how the unavailability of DeCSS could in itself convince millions of people to reject free operating systems, but I don't think I specifically mentioned about Corrupt Disks. I would have mentioned them if I had suspected I'd be given one.

Chavez is greatly admired by Venezuelans, who praise him in more glowing terms than I would praise anyone (even the people I most admire)

I handed the disk back to the person who had invited me, saying "Here you see the face of the enemy. Please bring this back to the store and get your money back, so they won't get any profit from this - and please don't buy Corrupt Disks!"

Copyright information

© 2005 by Richard Stallman

Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Richard Stallman is the founder of the GNU Project, launched in 1984 to develop the free software operating system GNU. The name "GNU" is a recursive acronym for "GNU's Not Unix".

Open Source Labs

Pioneers in Open Source Solutions

"Open Source Labs, a division of VYOM Labs, is a centre of excellence providing World-Class Support, Consultancy and Training across the entire spectrum of Open Source Technologies. Open Source Labs enables organisations worldwide to gain maximum value from Open Source Software while reducing the total cost of ownership".

Open Source Support Services

We provide migration, installation and post installation support solutions for most Open Source Technologies :

- Linux and MySQL
- Software Engineering tools like CVS, BugZilla
- Software IT management tools like Nagios, MRTG, OTRS
- Messaging solutions including Qmail, Sendmail, Webmail, SpamAssasin, Thunderbird
- Custom Open Source development in PHP, Perl (using MySQL or Postgres SQL servers)

Linux Training

- Linux OS Administration
- Networking and Security
- RedHat Linux
- SuSe Linux

Developer Course on Linux

- OSL 101 Unix Shell scripting
- OSL 102 PHP/Perl + MySQL

Open Source Labs
power of freedom

redhat®
TRAINING PARTNER
Authorized Red Hat Training Partner

vyomlabs
Looking beyond the Horizon

Open Source Labs, (A division of VYOM labs Pvt. Ltd.), Dayaprabha House, Opp Sulzer India, ITI Road, Aundh, Pune 411007, INDIA. Ph: +91 982278700, +91 20 2588 9236
e- mail : contactus@opensource-labs.com

www.opensource-labs.com

Paradigm Plus

Freelock Computing

The Open Source for Business Solutions

Open Source is about solving problems. We solve business problems for small and growing businesses in the Puget Sound area. We provide technology strategy consulting, implementation, and hosting. We work with proven open source software. Go to freelock.com for details.

"In today's world of crappy software vendors who provide crappy products and next to zero service at premium prices, it's refreshing to work with someone who is honest, thorough, reasonable and willing to do what it takes to meet the customer's needs."

– Eric Leung, IS Manager, Outdoor Research, Inc.

We Wrote The Book!

"'Open Source Solutions for Small Business Problems' is a very highly recommended book for anyone who is looking at the open source market ... This is easily one of the best Linux books of the year; providing a management level view of the Linux world without the technical focus of other books."

– Harold McFarland, Readers Preference Reviews

Find an open source service provider or sign up as one!
http://opensourceproviders.com

Contact us at:
info@freelock.com
1-206-579-4836

Freelock LLC
www.freelock.com

Free Software MAGAZINE

BY SUBSCRIBING YOU WILL BE SUPPORTING A MAGAZINE WHICH BELIEVES IN FREE SOFTWARE. ALL OUR ARTICLES ARE RELEASED UNDER THE GNU FREE DOCUMENTATION LICENSE, ENHANCING EXISTING INFORMATION ON FREE SOFTWARE.

SUBSCRIBE NOW!

HTTP://WWW.FREESOFTWAREMAGAZINE.COM/SUBSCRIBE